GA-Fuzzy Approaches: Application to Modeling of Manufacturing Process

Article · September 2013

DOI: 10.1007/978-3-642-25859-6_4

1 author:

Arup Kumar Nandi
Central Mechanical Engineering Research Institute

4

GA-Fuzzy Approaches: Application to Modeling of Manufacturing Process

Arup Kumar Nandi

Central Mechanical Engineering Research Institute (CSIR-CMERI)
Durgapur-713209, West Bengal, India
nandiarup@yahoo.com, nandi@cmeri.res.in

This chapter presents various techniques using the combination of fuzzy logic and genetic algorithm (GA) to construct model of a physical process including manufacturing process. First, an overview on the fundamentals of fuzzy logic and fuzzy inferences systems toward formulating a rule-based model (called fuzzy rule based model, FRBM) is presented. After that, the working principle of a GA is discussed and later, how GA can be combined with fuzzy logic to design the optimal knowledge base of FRBM of a process is presented. Results of few case studies of modeling various manufacturing processes using GA-fuzzy approaches conducted by the author are presented.

4.1 Introduction

Optimal selection of machining parameters is an imperative issue to obtain a better performance of machining, cost effectiveness as well as to achieve a desired accuracy of the attributes of size, shape and surface roughness of the finished product. Selection of these parameters is traditionally carried out on the basis of the experience of process planners with the help of past data available in machining handbooks and tool catalogs. Practitioners continue to experience great difficulties due to the lack of sufficient data on the numerous new cutting tools with different materials. Specific data on relevant machining performance measures such as tool life, surface roughness, chip form, etc. are very difficult to find due to the lack of reliable information or predictive models for these measures. In automated manufacturing processes, it is required to control the machining process by determining the optimum values of machining parameters online during machining. Therefore, it is important to develop a technique to predict the attribute of a product before machining to evaluate the robustness of machining parameters for keeping a desired attribute and increasing product quality. Construction of suitable machining process model and evaluation of the optimal values of machining parameters using this model as predictor are essential and challenging tasks.

The model of a machining process represents a mapping of input and output variables in specific machining conditions. The input variables differ corresponding to the type of machining process and the desired output. For example, in turning, the surface roughness (output variable) is dependent on a number of variables that can be broadly divided into four groups: major variables which include cutting speed, feed rate, depth of cut and tool wear; flow of coolant, utilization of chip breaker, work-holding devices and selection of tool type belong to the second group. The third group includes machine repeatability, machine vibration and damping, cutting temperature, chip formation and chip exit speed, thermal expansion of machine tool and power consumption; room temperature, humidity, dust content in the air and fluctuation in the power source are involved in the fourth group. Among these four groups of input variable, the major variable can be measured and controlled during machining process. Though the other variables are not directly involved and uncontrolled during machining but their effect cannot be neglected to obtain a desired surface roughness. In a specific machining condition, these variables are assumed to be fixed at a particular state.

Various approaches have been proposed to model and simulate the machining processes. Analytical methods, which are generally based on the established science principles, are probably the first modeling approach. Experimental or empirical approaches use experimental data as the basis for formulating the models. Mechanistic and numerical methods integrate the analytical and empirical methods, generally by the use of modern computer techniques.

Due to the complex and nonlinear relationship among the input–output variables, influence of uncontrollable parameters and involvement of random aspect, prediction of an output of machining processes using mathematical/analytical approaches is not accurate. This leads to the development of empirical equations for a particular machine tool, machining parameters and work piece-cutting tool material combination. Empirical models do not consider the underlying principles and mathematical relationships. These are usually obtained by performing statistical analysis or through the training of data-driven models to fit the experimental data [1].

The significant drawback of empirical models is their sensitivity to process variation though they have the advantages of accuracy due to the use of experimental data. The accuracy of a model degenerates rapidly due to the variation of experimental data as the machining conditions deviate from the experimental settings. In addition, quality characteristics of machined parts exhibit stochastic variations over time due to changes in a machine tool structure and the environment. Therefore, the modeling techniques should have enough capability to adapt the variations in machining process. Most of the statistical process control models do not account for time-varying changes. Involvement of uncertainty and imprecision in machining processes is another aspect affecting the variation of machining output. In such cases, techniques of modeling using fuzzy logic are most useful because fuzzy logic is a powerful tool for dealing with imprecision and uncertainty [2]. The basic concept of fuzzy logic is to categorize the variables into fuzzy sets with a degree of certainty in the numerical interval (0 and 1), so that ambiguity and vagueness in the data structure and human knowledge can be handled without constructing complex mathematical models. Moreover, fuzzy logic-based control system has the capability to adapt the variations of a process by learning and adjusting itself to the environmental changes by observing current system behavior.

Fuzzy logic is an application of fuzzy set theory, and was first proposed by Prof. L.A. Zadeh [3]. Fuzzy logic rules, which are derived based on fuzzy set theory, are used in fuzzy inference system toward formulating a rule-based model (called fuzzy rule-based model, FRBM). The performance of a FRBM mainly depends on two different aspects: structure of fuzzy logic rules and the type/shape of associated fuzzy subsets (membership function distributions, MFDs) those constitute the knowledge base (KB) of FRBM. Manually constructed KB of a FRBM may not be optimal in many cases since it strongly demands a thorough knowledge of the process which is difficult to acquire, particularly in a short period of time. Therefore, design of an optimal KB of a fuzzy model needs the help of other optimization/learning techniques. Genetic algorithm (GA), a population-based search and optimization technique is used by many researchers to design the optimal KB of FRBM for various processes. The systems of combining Fuzzy logic and genetic algorithm are called genetic-fuzzy systems.

4.2 Fuzzy Logic

4.2.1 Crisp Set and Fuzzy Set

A set (A) is a collection of any objects (a_1, a_2, a_3, ….., a_n), which according to some law can be considered as a whole and it is usually written as

$A=\{a_1, a_2,\ldots..,a_n\}$ or

$A=\{x \mid P(x)\}$, means A is the set of all elements (x) of universal set X for which the proposition P(x) is true (e.g. P(x) > 3).

In crisp set, the function $X_A(x)$ (so-called characteristic function) assigns a value of either 1 or 0 to each individual object, x in the universal set, thereby discriminating between members and non-members of the crisp set, A under consideration. That means there exists no uncertainty or vagueness in the fact that the object belongs to the set or does not belong to the set. Set A is defined by its characteristic function, $X_A(x)$ as follows

$$X_A(x) = \begin{cases} 1 \text{ for } x \in A \\ 0 \text{ for } x \notin A \end{cases}$$

In the year, 1965, Lotfi A. Zadeh [3] proposed a completely new and elegant approach to vagueness and uncertainty in a seminal paper, called *fuzzy set theory*. In his approach an element, x can belong to a set to a degree, k ($0 \le k \le 1$) in contrast to classical set theory where an element must definitely belong or not to a set. A fuzzy set, Ã is usually written as

$\tilde{A}=\{x, \mu_{\tilde{A}}(x) \mid x \in X\}$

The function $\mu_{\tilde{A}}(x)$ is called membership function (MF) of the (fuzzy) set $\tilde{A}$ and defined as $\mu_{\tilde{A}}(x) \rightarrow [0,1]$. The value of $\mu_{\tilde{A}}(x)$ is called the degree of membership of x in the fuzzy set $\tilde{A}$.

4.2.2 *Fuzzy Membership Function*

Graphically, a membership function is represented as a curve (as shown in Figure 4.1) that defines how each element in the set is mapped to a membership value (or degree of membership) between 0 and 1. There are many ways to assign membership values or functions to fuzzy variables compared to that of assigning probability density function to random variables. The membership function assignment process can be intuitive or based on some algorithmic or logical operations.

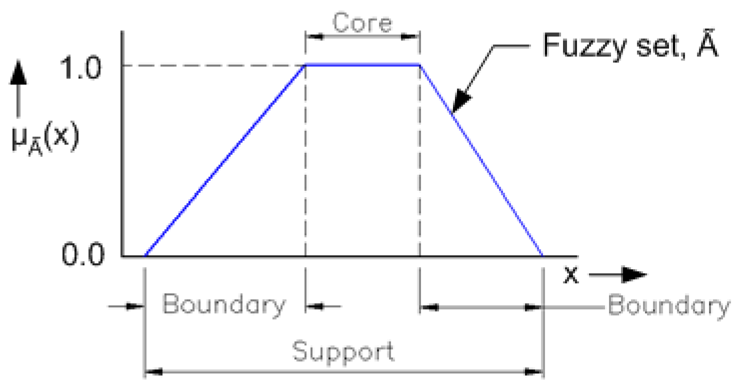

Fig. 4.1. A graphical representation of fuzzy set

4.2.2.1 Some Key Properties of Fuzzy Set

i) Having two fuzzy sets $\tilde{A}$ and $\tilde{B}$ based on X, then both are *equal* if their membership functions are equal, i.e. $\tilde{A} = \tilde{B} \Leftrightarrow \mu_{\tilde{A}}(x) = \mu_{\tilde{B}}(x)$ for all $x \in X$

ii) Given a fuzzy set $\tilde{A}$ defined on X and any number $\alpha \in [0,1]$, the *α-cut*, ${}^{\alpha}\tilde{A}$, and the *strong α-cut*, ${}^{\alpha+}\tilde{A}$, are the crisp sets: ${}^{\alpha}\tilde{A} = \{x \mid \tilde{A}(x) \geq \alpha\}$ and ${}^{\alpha+}\tilde{A} = \{x \mid \tilde{A}(x) > \alpha\}$

iii) The *height* of a fuzzy set is the largest membership grade obtained by any element in that set i.e., $\text{height}(\tilde{A}) = \max_{x \in X} \mu_{\tilde{A}}(x)$

iv) The *crossover points* of a membership function are defined as the elements in the universe for which a particular fuzzy set A has values equal to 0.5, i.e., for which $\mu_{\tilde{A}}(x) = 0.5$

v) A fuzzy set $\tilde{A}$ is called *normal* when $\text{height}(\tilde{A}) = 1$ and *subnormal* when $\text{height}(\tilde{A}) < 1$.

vi) The *support* of a fuzzy set $\tilde{A}$ is the crisp set that contains all the elements of X that have non-zero membership grades, i.e. $\text{support}(\tilde{A}) = \{x \in X \mid \mu_{\tilde{A}}(x) > 0\}$, refer to Figure 4.1.

vii) The *core* of a normal fuzzy set $\tilde{A}$ is the crisp set that contains all the elements of X that have the membership grades of one in $\tilde{A}$, i.e. $\text{core}(\tilde{A}) = \{x \in X \mid \mu_{\tilde{A}}(x) = 1\}$, refer to Figure 4.1.

viii) The *boundary* is the crisp set that contains all the elements of X that have the membership grades of $0 < \mu_{\tilde{A}}(x) < 1$ in $\tilde{A}$, i.e. $\text{boundary}(\tilde{A}) = \{x \in X \mid 0 < \mu_{\tilde{A}}(x) < 1\}$, refer to Figure 1.

ix) Having two fuzzy sets $\tilde{A}$ and $\tilde{B}$ based on X, then both are *similar* if $\text{core}(\tilde{A}) = \text{core}(\tilde{B})$ and $\text{support}(\tilde{A}) = \text{support}(\tilde{B})$.

x) If the support of a normal fuzzy set consists of a single element x_o of X, which has the property $\text{support}(\tilde{A}) = \text{core}(\tilde{A}) = \{x_0\}$, this set is called a *singleton*.

xi) A fuzzy set $\tilde{A}$ is said to be a *convex* fuzzy set if for any elements x, y and z in fuzzy set $\tilde{A}$, the relation x<y<z implies that $\mu_{\tilde{A}}(y) \geq \min[\mu_{\tilde{A}}(x), \mu_{\tilde{A}}(z)]$. The intersection of two convex fuzzy sets is also a convex fuzzy set, i.e., if $\tilde{A}$ and $\tilde{B}$ are *convex* fuzzy sets, then $\tilde{A} \cap \tilde{B}$ is also convex fuzzy set.

xii) If $\tilde{A}$ is convex single-point normal fuzzy set defined on the real line, then $\tilde{A}$ is often termed as a *fuzzy number*.

xiii) Any fuzzy set $\tilde{A}$ defined on a universe X is a *subset* of that universe.

4.2.2.2 Various Types of Fuzzy Membership Function and Its Mathematical Representation

The common types of membership function (MF) used in FRBM are triangular, (higher order) polynomial, trapezoidal, Gaussian, etc.

Trapezoidal MF: Mathematically a trapezoidal MF can be represented as shown in Figure 4.2(a).

$$\mu_{\tilde{A}}(x,a,b,c,d) = \begin{matrix} 0 \\ \dfrac{x-a}{b-a} \\ 1 \\ \dfrac{d-x}{d-c} \end{matrix} \quad \text{for} \begin{cases} x < a, x > d \\ a \leq x \leq b \\ b < x < c \\ c \leq x \leq d \end{cases}$$

The controlling parameters toward the configuration of trapezoidal MF (as shown in Figure 4.2(a)) are b_1=b-a, b_2=c-b and b_3=d-c.

Polynomial MF: A polynomial MF can be expressed mathematically as shown in Figure 4.2(b):

$$\mu_{\tilde{A}}(x,a,b,c,d) = \begin{matrix} 0 \\ f_1(x,b_1) \\ 1 \\ f_2(x,b_2) \end{matrix} \quad \text{for} \begin{cases} x < a, x > c \\ a \le x < b \\ x = b \\ b < x \le c \end{cases}$$

where the functions f_1 and f_2 are the polynomial type. Polynomial MF is treated as triangular type MF when the functions, f_1 and f_2 of the above empirical expression are linear. f_1 and f_2 may be also exponential or any other kind of functions.

The controlling parameters toward the configuration of polynomial MF (as shown in Figure 4.2(b)) are b_1=b-a and b_2=c-b. Mathematically, the second order polynomial function can be represented as $\mu_{\tilde{A}}(x) = c_0 x + c_1 x^2$, where x is the distance measured along the base-width of membership function distributions, $\mu_{\tilde{A}}(x)$ is the fuzzy membership function value and, c_0 and c_1 are the coefficients which can be determined based on some specified conditions, such as

$$\mu_{\tilde{A}} = \begin{cases} 1 \text{ at } x = b_1 \\ 0 \text{ at } x = 0 \end{cases} \text{ and } \frac{\partial \mu_{\tilde{A}}}{\partial x} = 0, \text{at } x = b_1,$$

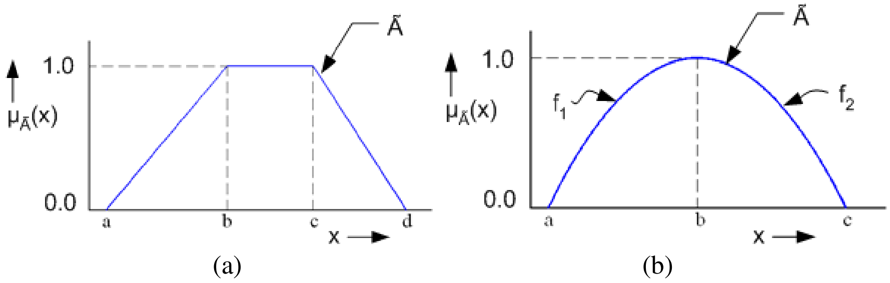

Fig. 4.2. Membership function configuration (a) Trapezoidal (b) Polynomial

Finally the coefficients of the 2nd order polynomial function become $c_0 = \frac{2}{b_1}$ and $c_1 = -\frac{1}{b_1^2}$.

From Figure 4.2, it has been seen that for a given value of support of the membership function of a fuzzy set, only one parameter, b_1 is required to describe triangular and polynomial type MFDs (membership function distributions) with 3 fuzzy sub-sets, whereas two parameters, b_1 and b_2 are required to explain the (semi) trapezoidal MFDs with two fuzzy subsets. The number of controlling parameter increases with increasing the number of fuzzy sub-sets involved in the MFDs.

4.2.3 Fuzzy Set Operators

Defining the fuzzy sets $\tilde{A}$ and $\tilde{B}$ on the universe X, for a given element x of the universe, the fuzzy set operations, intersection (t-norm), union (t-conorm) and complement are expressed as follows and the corresponding Venn diagrams are shown in Figure 4.3.

Intersection: $\mu_{\tilde{A}\cap\tilde{B}}(x) = \mu_{\tilde{A}}(x) \wedge \mu_{\tilde{B}}(x)$

Union: $\mu_{\tilde{A}\cup\tilde{B}}(x) = \mu_{\tilde{A}}(x) \vee \mu_{\tilde{B}}(x)$

Complement: $\mu_{\bar{\tilde{A}}}(x) = 1 - \mu_{\tilde{A}}(x)$

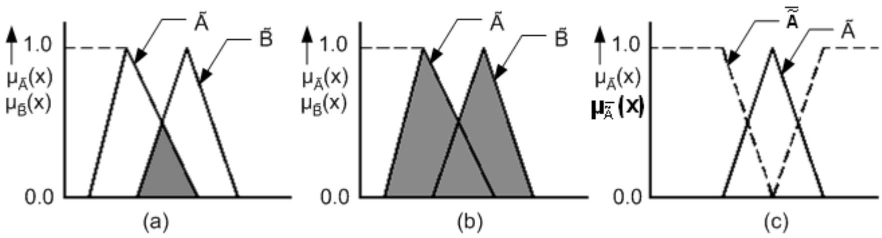

Fig. 4.3. (a) Intersection of fuzzy sets $\tilde{A}$ and $\tilde{B}$ (b) Union of fuzzy sets $\tilde{A}$ and $\tilde{B}$ (c) Complement of fuzzy set $\tilde{A}$

4.2.4 Classical Logical Operations and Fuzzy Logical Operations

Classical logic deals with classical proposition (P) which is a collection of elements, that is, a set, where all the truth values, T(P) for all elements in the set are either all true (1) or all false (0), and follows the two-valued logical operations and Boolean algebra.

Let the sets A and B are defined from universe X and the proposition, P measures the truth of the statement that an element, x from the universe X is contained in set, A and Q measures the truth of the statement that this element, x is contained in set, B, i.e., if $x \in A, T(P)=1;$ otherwiase $T(P)=0$ and if $x \in B, T(Q)=1;$ otherwise $T(Q)=0$. There are five logical connectives (defined as follows) used to combine multiple simple propositions and to form new propositions:

Disjunction (OR):

$$P \vee Q : x \in A \text{ or } x \in B, \text{Hence}, T(P \vee Q) = \max(T(P), T(Q))$$

Conjunction (AND):

$$P \wedge Q : x \in A \text{ or } x \in B, \text{Hence}, T(P \wedge Q) = \min(T(P), T(Q))$$

Negation: $\text{If } T(P) = 1, \text{ then } T(\overline{P}) = 0; \text{if } T(P) = 0, \text{ then } T(\overline{P}) = 1$

Implication: $P \rightarrow Q : x \notin A \text{ or } x \in B, \text{Hence}, T(P \rightarrow Q) = T(\overline{P} \cup Q)$

Equivalence: $(P \leftrightarrow Q) : T(P \leftrightarrow Q) = \begin{cases} 1 \text{ for } T(P) = T(Q) \\ 0, \text{for } T(P) \neq T(Q) \end{cases}$

For two different universes of discourse where P is a proposition described by set A, which is defined on universe X, and Q is a proposition described by set B, which is defined on universe Y. Then the implication P→Q (which is also equivalent to the linguistic rule form, IF A THEN B) can be represented in set-theoretic terms by the relation, R which is defined by

$$R = (A \times B) \cup (\overline{A} \times Y) \equiv \text{IF A, THEN B}$$
$$\text{IF } x \in A, \text{ where } x \in X \text{ and } A \subset X$$
$$\text{THEN } y \in B, \text{ where } y \in Y \text{ and } B \subset Y$$

The other connectives are applicable to two different universes of discourse as usual. Classical logical compound propositions that are always true irrespective of the truth values of the individual simple propositions are called tautologies.

Fuzzy propositional logic generalizes the classical propositional operations by using the truth set [0, 1] instead of either 1 or 0. The above logical connectives are also defined for a fuzzy logic. Like classical logic, the implication connective in fuzzy logic can be modeled in rule-based form: $\tilde{P} \rightarrow \tilde{Q}$ is, IF x is $\tilde{A}$ THEN y is $\tilde{B}$ (where IF part is called *antecedent* and THEN part is called *consequent*) and it is equivalent to the fuzzy relation $\tilde{R} = (\tilde{A} \times \tilde{B}) \cup (\overline{\tilde{A}} \times Y)$ where the fuzzy proposition $\tilde{P}$ is assigned to fuzzy set $\tilde{A}$ which is defined on universe X, and the fuzzy proposition $\tilde{Q}$ is described by fuzzy set $\tilde{B}$, which is defined on universe Y. The membership function of $\tilde{R}$ is expressed by $\mu_{\tilde{R}}(x, y) = \max[(\mu_{\tilde{A}}(x) \wedge \mu_{\tilde{B}}(y)), (1 - \mu_{\tilde{A}}(x))]$. The implication connective can be defined in several distinct forms. While these forms of implication are equivalent in classical logic, their extensions to fuzzy logic are not equivalent and result in distinct classes of fuzzy implications.

4.2.5 Fuzzy Implication Methods

The fuzzy implication operation is used to find the fuzzy relation $\tilde{R}$ between two fuzzy sets $\tilde{A}$ and $\tilde{B}$ which are defined on the universes of discourse X and Y, respectively based on the rule IF x is $\tilde{A}$, THEN y is $\tilde{B}$, where $x \in X$ and $y \in Y$. Mathematically, the fuzzy relation, $\tilde{R}$ is defined as $\tilde{R}(x, y) = \wp[A(x), B(y)]$, where $\wp$ is called the implication operator. Besides the implication method as presented in Section 4.2.4, there are other forms of implication operator, among them *min* and *product* implication operators are mostly used in fuzzy inference system for practical applications. The membership function values of fuzzy relation $\tilde{R}$ defined on the Cartesian product space $X \times Y$ using min and product implication operators are obtained by the Equation (4.1) and Equation (4.2), respectively and graphically represented in Figure 4.4.

min: $$\mu_{\tilde{R}}(x, y) = \min\left[\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(y)\right] \quad (4.1)$$

product: $$\mu_{\tilde{R}}(x, y) = \mu_{\tilde{A}}(x) \bullet \mu_{\tilde{B}}(y) \quad (4.2)$$

In Figure 4.4, the MF value, 0.7 of $\mu_{\tilde{B}}(x)$ corresponds to rule weight obtained after decomposition of the IF part of rule.

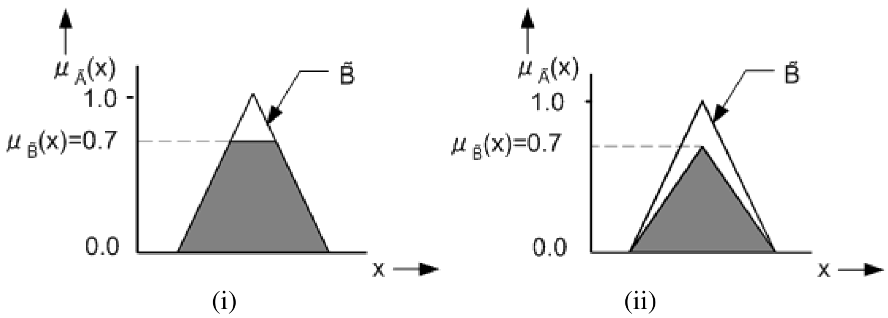

Fig. 4.4. Graphical representation of fuzzy implication (i) min (ii) product

4.2.6 Decomposition of Compound Rules

The most common techniques for decomposition of compound linguistic rules into simple canonical forms are described as follows:

Multiple conjunctive antecedents: IF x is $\tilde{A}_1$ AND x is $\tilde{A}_2$ …. AND x is $\tilde{A}_L$ THEN y is $\tilde{B}_S$

This fuzzy rule can be written in canonical form as: IF $\tilde{A}_S$ THEN $\tilde{B}_S$, where the fuzzy subset $\tilde{A}_S$ is defined as $\tilde{A}_S = \tilde{A}_1 \cap \tilde{A}_2 \cap ...... \cap \tilde{A}_L$) with the membership function $\mu_{\tilde{A}_S}(x) = \min\lfloor \mu_{\tilde{A}_1}(x), \mu_{\tilde{A}_2}(x), ..., \mu_{\tilde{A}_L}(x) \rfloor$ obtained by the definition of the fuzzy intersection operation.

Multiple disjunctive antecedents: IF x is $\tilde{A}_1$ OR x is $\tilde{A}_2$ …. OR x is $\tilde{A}_L$ THEN y is $\tilde{B}_S$

Assuming a new fuzzy subset $\tilde{A}_S$ (as $\tilde{A}_S = \tilde{A}_1 \cup \tilde{A}_2 \cup ...... \cup \tilde{A}_L$) expressed by means of membership function $\mu_{\tilde{A}_S}(x) = \max\lfloor \mu_{\tilde{A}_1}(x), \mu_{\tilde{A}_2}(x), ..., \mu_{\tilde{A}_L}(x) \rfloor$ based on the definition of the fuzzy union operation, the compound rule may be written in canonical form as: IF $\tilde{A}_S$ THEN $\tilde{B}_S$.

4.2.7 Aggregation of Rule

The technique of obtaining the overall rule consequent by combining the individual consequents contributed by each rule in the rule base (which comprises multiple rules) is known as aggregation of rules. The most popular aggregation techniques of fuzzy rules are as follows:

Conjunctive system of rules (MIN): In the case of a system of rules that must be jointly satisfied, the rules are connected by AND connectives. In this case, the aggregated consequent, y is obtained by fuzzy intersection of all individual rule consequents (y_1, y_2, …, y_r), as $y = y_1 \cap y_2 \cap ...... \cap y_r$ which is defined by the membership function: $\mu_y(y) = \min\lfloor \mu_{y_1}(y), \mu_{y_2}(y), ..., \mu_{y_r}(y) \rfloor$ for $y \in Y$.

Disjunctive system of rules (MAX): For the case of a disjunctive system of rules where the satisfaction of at least one rule is required, the rules are connected by the OR connectives. In this case, the aggregated consequent, y is obtained by fuzzy union of the individual rule consequents as $y = y_1 \cup y_2 \cup ...... \cup y_r$ which is defined by the membership function: $\mu_y(y) = \max\lfloor \mu_{y_1}(y), \mu_{y_2}(y), ..., \mu_{y_r}(y) \rfloor$ for $y \in Y$.

4.2.8 Composition Technique of Fuzzy Relation

Let R and S be the relations that relate elements from universe X to universe Y, and elements from universe Y to universe Z, respectively. Now, composition is an operation to find another relation, T that relates the same elements in universe X that R contains to the same elements in universe Z that S contains. The composition operation of fuzzy relation reflects the inference of a fuzzy rule-based system and is expressed by $\tilde{B} = \tilde{A} \circ \tilde{R}$, where $\tilde{A}$ is the input, or antecedent defined on the universe X, $\tilde{B}$ is the output or consequent defined on universe Y and $\tilde{R}$ is a fuzzy relation characterizing the relationship between specific input(s), x and specific output(s), y. Among various methods of composition of fuzzy relation, max-min and max-product are the most commonly used techniques and defined by membership function-theoretic expressions as follows.

$$\text{max-min: } \mu_{\tilde{B}}(y) = \max_{x \in X} \left\{ \min\left[\mu_{\tilde{A}}(x), \mu_{\tilde{R}}(x, y) \right] \right\} \tag{4.3}$$

$$\text{max-product: } \mu_{\tilde{B}}(y) = \max_{x \in X} \left[\mu_{\tilde{A}}(x) \bullet \mu_{\tilde{R}}(x, y) \right] \tag{4.4}$$

The method of composition of fuzzy relation basically includes the implication method and technique of aggregation of fuzzy rule. In the above methods of composition of fuzzy relation, max is the aggregation technique of rule, whereas min and product are the implication methods used in Equation (4.3) and Equation (4.4), respectively.

4.2.9 Fuzzy Inferences

Inference is a process of combining the measurement of input(s)/antecedent(s) with one or more relevant fuzzy rules in a proper manner to infer the output(s)/consequent(s). In order to demonstrate the inference method, consider a system of one input (antecedent) and a single output (antecedent) described by two IF-THEN rules as follows:

Rule 1: if input1 is $\tilde{A}_1$, then output1 is $\tilde{B}_1$
Rule 1: if input1 is $\tilde{A}_2$, then output1 is $\tilde{B}_2$

Now given the IF-THEN rules (Rule 1 and Rule 2) and a fact/measurement "Input1 is $\tilde{A}$" it is inferred that "output1 is $\tilde{B}$", where $\tilde{A}$, $\tilde{A}_1 \in \mathcal{F}(\chi)$ and $\tilde{B}$, $\tilde{B}_1 \in \mathcal{F}(\Upsilon)$, where $\mathcal{F}(\chi)$ and $\mathcal{F}(\Upsilon)$ denote the sets of all ordinary fuzzy sets that can be defined within the X and Y, respectively. χ and Υ are the sets of values (x and y) of variables, input1 (condition variable) and output1 (action variable), respectively. In order to determine B, method of interpolation is used which consists of two steps explained as follows:

Step 1: Calculate the *degree of consistence,* $r_j(x)$ between the given fact/measurement and the antecedent of each rule, j in terms of the *height* of intersection of the associated sets $\tilde{A}_1$ and $\tilde{A}$. $r_j(x)$ is expressed using the standard fuzzy intersection and the definition of height of a fuzzy set (described in Section 4.2.2.1) as follows:

$$r_j(x) = \max_{x \in X} \min\left[\mu_{\tilde{A}}(x), \mu_{\tilde{A}_j}(x) \right], \ j=1, 2 \tag{4.5}$$

Step 2: Calculate the conclusion $\tilde{B}$ by truncating each set $\tilde{B}_j$ by the value of $r_j(x)$ (i.e., *min* implication method) which expresses the degree to which the antecedent $\tilde{A}_j$ is compatible with the given fact $\tilde{A}_1$, and taking the union of the truncated sets as the rules are satisfied independently (i.e., *max* aggregation method) .

$$\mu_{\tilde{B}}(y) = \max_{j=1,2} \min\left[r_j(x), \mu_{\tilde{B}_j}(y)\right] \text{ for all } y \in Y \tag{4.6}$$

The above steps are graphically presented in Figure 4.5.

$$\text{Now, } \mu_{\tilde{B}}(y) = \max_{j=1,2} \min\left[r_j(x), \mu_{\tilde{B}_j}(y)\right]$$

$$= \max_{j=1,2} \min\left[\max_{x \in X} \min\left(\mu_{\tilde{A}}(x), \mu_{\tilde{A}_j}(x)\right), \mu_{\tilde{B}_j}(y)\right]$$

$$= \max_{j=1,2} \max_{x \in X}\left[\min\left(\mu_{\tilde{A}}(x), \mu_{\tilde{A}_j}(x)\right), \mu_{\tilde{B}_j}(y)\right]$$

$$= \max_{x \in X} \max_{y=1,2} \min\left[\mu_{\tilde{A}}(x), \min\left(\mu_{\tilde{A}_j}(x), \mu_{\tilde{B}_j}(y)\right)\right]$$

$$= \max_{x \in X} \min\left[\mu_{\tilde{A}}(x), \max_{j=1,2} \min\left(\mu_{\tilde{A}_j}(x), \mu_{\tilde{B}_j}(y)\right)\right]$$

$$= \max_{x \in X} \min\left[\mu_{\tilde{A}}(x), \mu_{\tilde{R}}(x, y)\right], \tag{4.7}$$

which is equivalent to the expression of max-min composition presented in Equation (4.3) and accordingly, this inference method is also called max-min inference method. In the above step 2, if *product* implication technique and *max* aggregation method are used, we can obtain the expression (defined in Equation 4.7) similar to max-product composition (Equation (4.4)). Moreover, the above inference method may also be applicable to the system with any number of fuzzy rule (j) and inputs (Input1, input2). In case of multiple inputs/antecedents of fuzzy rule, the (effective) *degree of consistence,* $r_j(x)$ between the given facts/measurements and the antecedents of each rule is obtained by first finding the *degree of consistence* between each fact/measurement and the related antecedent of the rule (using Equation (4.5)), then adopting the technique of decomposition of compound rules according to the type of logical connectives (AND or OR) as explained graphically in Figure 4.6 for the following two IF-THEN rules as follows:

Rule 1: if input1 is $\tilde{A}_{11}$ AND input2 is $\tilde{A}_{12}$ then output1 is B_1
Rule 1: if input1 is $\tilde{A}_{21}$ AND input2 is $\tilde{A}_{22}$ then output1 is B_2
Fact: input1 is $\tilde{A}_1$ AND input2 is $\tilde{A}_2$
Conclusion: output1 is B

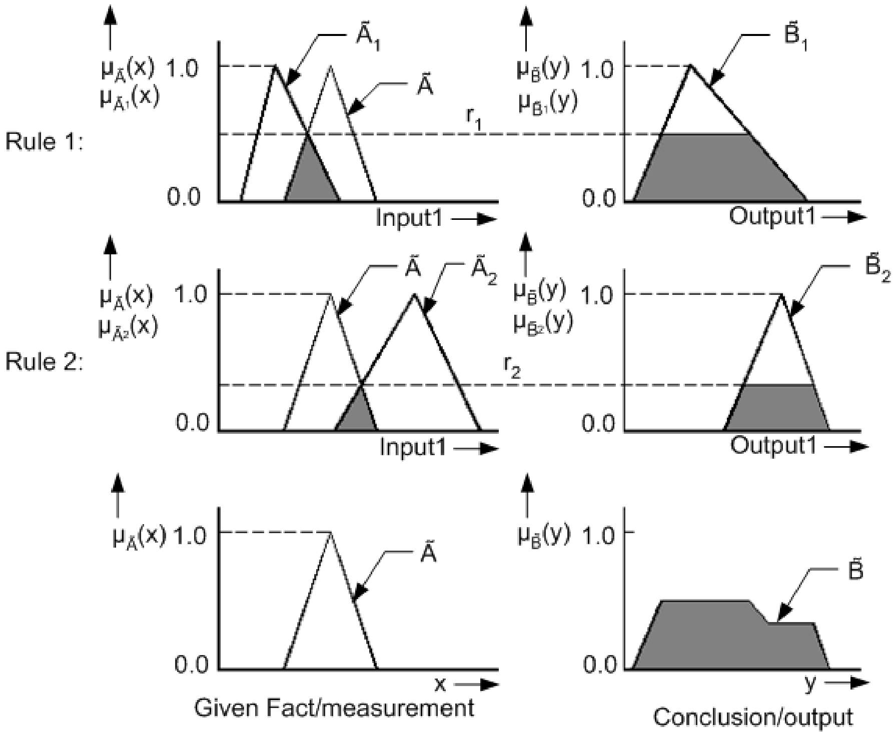

Fig. 4.5. Illustration of the method of interpolation in fuzzy inferences

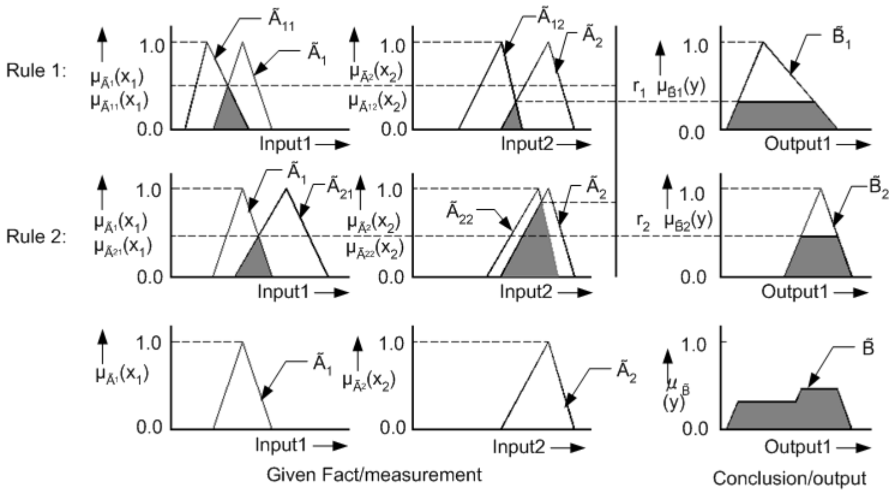

Fig. 4.6. Illustration of the method of interpolation in fuzzy inferences with multiple inputs

In the above illustration of fuzzy inference, we have considered fuzzy value ($\tilde{A}$) for the variable input1. Figure 4.7 demonstrates the above (max-min) inference method for a two input and a single output system where the values of input variables are considered as crisp type (for instance, Fact: input1 is x_1 AND input2 is x_2) and the (max-product) inference method for the same system is demonstrated in Figure 4.8.

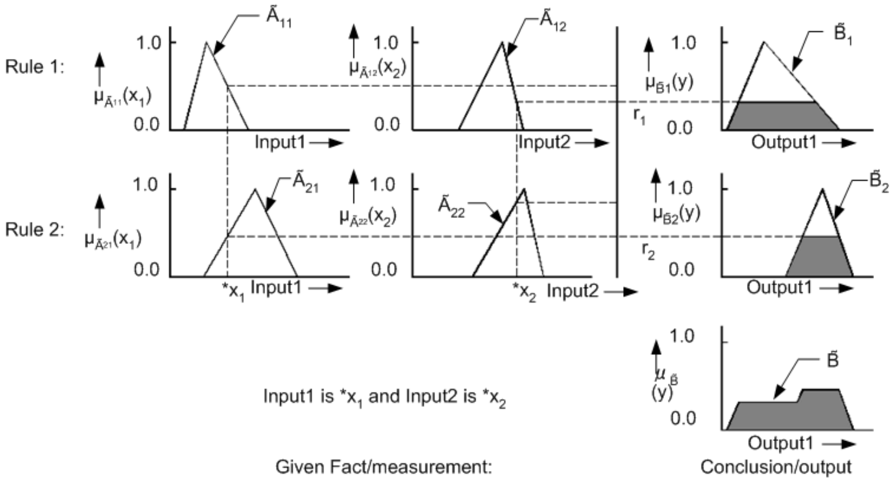

Fig. 4.7. Graphical representation of max-min inference method with crisp type of input values

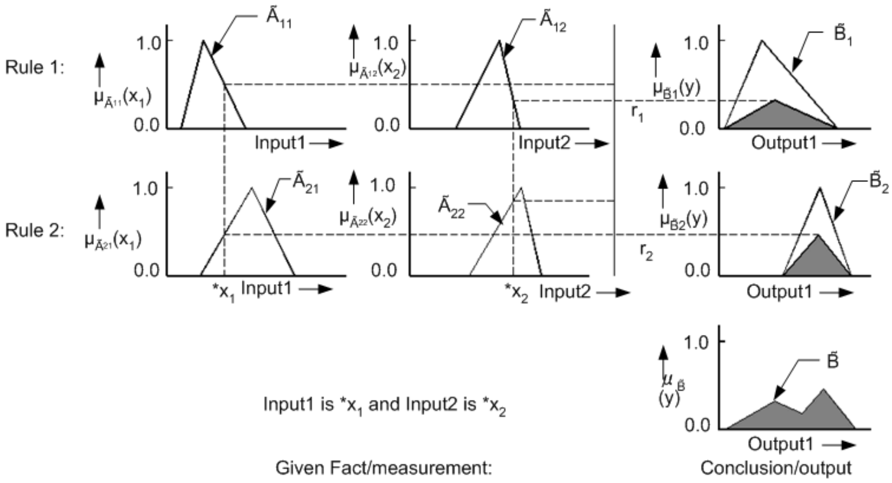

Fig. 4.8. Graphical representation of max-product inference method with crisp type of input values

4.2.10 Fuzzification and De-fuzzification

The fuzzification is a process of transforming the crisp value into a grade of membership using a membership function of the associated fuzzy set as shown in Figure 4.9. Figure 4.9 demonstrates that the given (crisp) value x_0 of variable x belongs to a grade of $\mu_{\tilde{A}_1}(x_0) = 0.8$ to the fuzzy set $\tilde{A}_1$ and with a grade of $\mu_{\tilde{A}_2}(x_0) = 0.2$ to the fuzzy set $\tilde{A}_2$. Fuzzification is required in the fuzzy inference system when the values of input variables to system are considered as crisp type (as described in Figure 4.7/Figure 4.8).

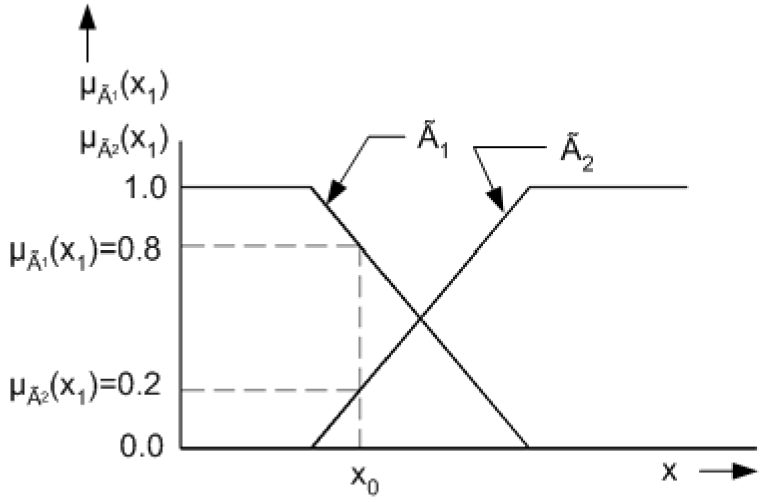

Fig. 4.9. Fuzzification method

Defuzzification is the conversion of a fuzzy quantity to a crisp quantity analogous to fuzzification. Defuzzification is used in fuzzy inference system to convert the fuzzy value of the output (i.e., $\tilde{B}$ of output1 in Section 4.2.9) to a crisp value (*y). Among various defuzzification methods available in the literature, centroid method (also called center of area (COA) or center of gravity) is most popular, which is mathematically expressed by Equation (4.8) and graphically expressed in Figure 4.10:

$$y_{COA} = \frac{\int \mu_{\tilde{B}}(y) \cdot y dy}{\int \mu_{\tilde{B}}(y) dy} \tag{4.8}$$

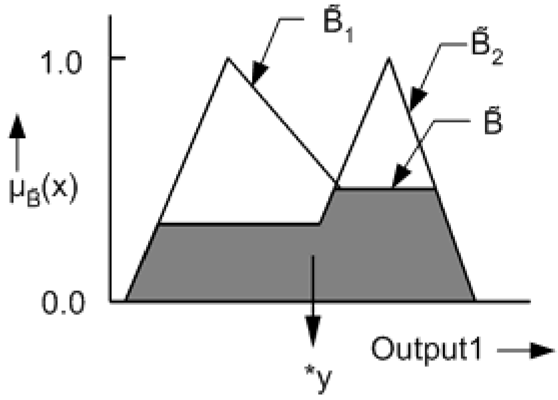

Fig. 4.10. Centroid defuzzification method

4.2.11 Fuzzy Rule-Based Model

4.2.11.1 Working Principle of a Fuzzy Rule-Based Model

A fuzzy rule base, a fuzzy inference engine, fuzzification and de-fuzzification: these are the four modules involved in a FRBM (Fuzzy rule-based model). Figure 4.11 shows a schematic diagram explaining the working cycle of a FRBM. The following steps are involved in the working cycle of a FRBM:

- The output(s) (action) and input(s) (condition) variables needed to control a particular process are chosen and measurements are taken for all the condition variables.
- The measurements taken in the previous steps are converted into appropriate fuzzy sets to express measurement uncertainties (called fuzzification as described in Section 4.2.10).
- The fuzzified measurements are then used by the interference engine to evaluate the control rules stored in the rule base and a fuzzified output is determined (as discussed in Section 4.2.9).
- The fuzzified output is then converted into a single (crisp) value (called a defuzzification as illustrated in Section 4.2.10). The de-fuzzified value(s) represent action(s)/prediction(s) to be made by the FRBM in controlling a process.

The kernel of a FRBM that is the knowledge base (KB) is constituted by rule base, RB (a set of fuzzy logic rules) and membership functions/membership function distributions, MFDs (fuzzy subsets). Two types of fuzzy logic rules (FLRs) that are commonly used for constructing the RB of a FRBM are Mamdani-type and TSK-type. In both types of FLRs, the input variables are expressed by linguistic terms (fuzzy subsets) in the rule antecedent part. But the main difference between these two types of fuzzy rules lies in the rule consequent part. The output variable in Mamdani-type FLR is defined by linguistic term also, whereas in TSK-type FLR, it is not defined by linguistic term rather it is defined by a linear combination of the input variables. The shape of fuzzy subsets (MFDs of input-output variables) is also an important factor that is to be decided appropriately to achieve the best performance of a FRBM for a typical process.

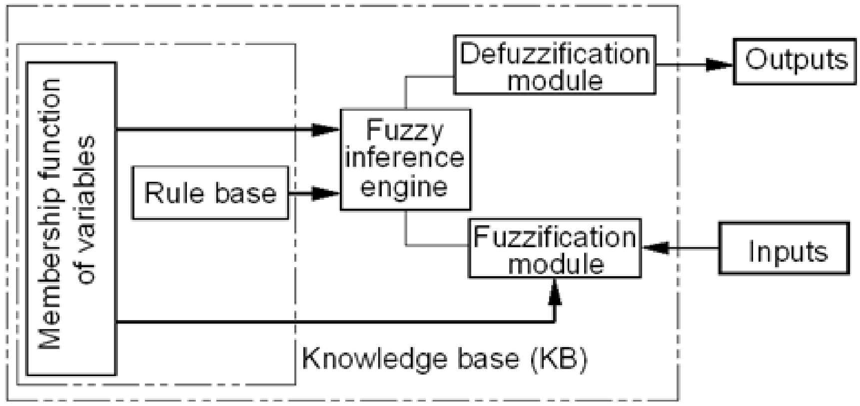

Fig. 4.11. A schematic showing the working cycle of a FRBM

4.2.11.2 Various Types of Fuzzy Rule-Based Model

Mamdani-type [4]:
The structure of Mamdani-type fuzzy logic rule is expressed as follows:

IF x_1 is A_1 AND x_2 is A_2 AND……..AND x_n is A_n THEN y is B

where x_i (i=1, 2, ……, n) are input variables and y is the output variable. A_1, A_2, …, A_n and B are the linguistic terms (say, Low, Medium, High, etc.) used for the fuzzy subsets (membership function distributions) of the corresponding input and output variables, respectively.

Sugeno-type [5]:
The Sugeno-type fuzzy rule is defined as follows:

IF x_1 is A_1 AND x_2 is A_2 AND……..AND x_n is A_n THEN $y = f(x_1, x_2, .., x_n)$

Unlike Mamdani-type, the rule consequent/output is expressed by a function of the input variables.

Tsukamoto [6]:
The Tsukamoto -type fuzzy rule is defined as follows:

IF x_1 is A_1 AND x_2 is A_2 AND……..AND x_n is A_n THEN y =z

where z is a monotonical membership function.

4.2.11.2.1 Mamdani-Type Fuzzy Rule Based Model

The output of a Mamdani-type FRBM whose rule base (RB) is constructed using Mamdani-type fuzzy logic rule is obtained as follows (when centroid method is considered for defuzzification):

$$Y = \frac{\sum_{r=1}^{R_f^1} A^{\alpha r} C_{A^{\alpha r}}}{\sum_{r=1}^{R_f^1} A^{\alpha r}} \tag{4.9}$$

where $A^{\alpha r}$ is the area of fuzzy subset of output variable, covered by α membership value (equivalent to *degree of consistence* as obtained in Step 1 in Section 4.2.9) that is obtained by r^{th} rule after fuzzy inference method. $C_{A^{\alpha r}}$ is the center distance of the area, $A^{\alpha r}$. R_f^1 ($R_f^1 \subseteq R_f$) is the number of rules fired out of a total of R_f rules present in the rule base for a set of input values.

In order to determine the output in Equation 4.9, all the four modules (fuzzy rule base, a fuzzy inference engine, fuzzification and de-fuzzification) as mentioned in Section 4.2.11.1 are involved in a Mamdani-type FRBM. The performance of a Mamdani-type fuzzy model is relied on the appropriate fuzzy subsets of rule consequents and antecedent, and the type of fuzzy subsets (membership function distributions) considered for input-output variables. Therefore, the tasks of designing FRBMs with Mamdani-type FLRs are:

- construction of an optimal set of rules (R_f) with its appropriate outputs (B),
- selection of shape of fuzzy subsets/MFDs for both the input and output variables
- tuning of MFDs.

4.2.11.2.2 TSK-Type Fuzzy Rule-Based Model

The TSK-type fuzzy logic rule is defined as follows [5, 7]:

If x_1 is A_1 and x_2 is A_2 and…………………and x_n is A_n, then $y = \sum_{j=1}^{K} c_j f_j(x_{1,\ldots,n})$

where $A_1, \ldots, A_n$ are the fuzzy subsets of the respective input variables, $x_1, \ldots, x_n$. The output function of fuzzy rule is a linear function (say, polynomial) in the form of

$$y = \sum_{j=1}^{K} c_j f_j(x_{1,\ldots,n}) = c_1 f_1(x_{1,\ldots,n}) + c_2 f_2(x_{1,\ldots,n}) + \ldots.. + c_k f_K(x_{1,\ldots,n}) \tag{4.10}$$

The overall output of the TSK-type fuzzy model can be obtained for a set of inputs ($x_1, x_2, \ldots., x_n$) using the following empirical expression.

$$Y = \frac{\sum_{r=1}^{R_f^1}\left(\prod_{v=1}^{n}\mu_v(x_v)\right)\sum_{j=1}^{K}c_j^r f_j^r(x_{1,\dots,n})}{\sum_{r=1}^{R_f^1}\left(\prod_{v=1}^{n}\mu_v^r(x_{1,\dots,n})\right)} \tag{4.11}$$

$\prod$ is the product representing a conjunction decomposition method. $\sum_{j=1}^{K}c_j^r f_j^r(x_{1,\dots,n})$ is the output function of r^{th} rule and c_j^r are the function coefficients of the corresponding rule consequent, where K is the number of coefficients present in the consequent function of each rule.

Unlike Mamdani-type FRBM, TSK-type FRBM includes only the fuzzy rule base, a fuzzy inference engine, and fuzzification module to determine the output in Equation (4.11). The performance of a TSK-type fuzzy model is mainly depended on the optimal values of the rule output (consequent) functions which are depended on the coefficients (c_j), the exponential parameters of the input variables (not shown in the Equation (4.10)) and choice of the fuzzy subsets (membership function distributions). Thus, the steps of developing FRBM with TSK-type FLRs are:

- construction of an optimal set of rules (R_f) with the appropriate structures of rule output/consequent functions
- selection of shapes of fuzzy subsets/MFDs of input variables
- determination of optimal values of coefficients and power terms of rule consequent functions
- tuning of MFDs of the input variables.

4.3 Genetic Algorithm

Genetic algorithm is a search and optimization technique which mimics the principle of natural selection and natural genetics [8] to find the best solution for a specific problem. The genetic algorithm is an approach to solve problems which are not yet fully characterized or too complex to allow full characterization, but for which some analytical evaluation or physical interpretation to evolutes the performance of a solution, is available. GA is a stochastic global search method that mimics the metaphor of natural biological evolution. Genetic algorithms operate on a population of feasible solutions by applying the principle of survival of the fittest to produce better approximations to a solution. At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the problem domain and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation.

The basic concept of a genetic algorithm is to encode a potential solution to a problem as a series of parameter strings, *chromosomes*, composed over some alphabet, so that chromosome values (*genotypic*) are uniquely mapped onto the decision variable (*phenotypic*). A single set of parameter string or chromosome is treated as the genetic material of an individual solution. Initially a large population of candidate solutions is created with random parameter values. These solutions are essentially bred with each other for several simulated generations under the principle of survival of the fittest, meaning that the probability that an individual solution will pass on some of its parameter values to subsequent children is directly related to the fitness of individual i.e. how good that solution is relative to the others in the population.

Breeding takes place through use of recombination operators such as crossover, which simulates basic biological cross-fertilization, and mutation, essentially the introduction of noise. The simple application of these operators with a reasonable selection mechanism has produced surprisingly good results over a wide range of problems. After recombination and mutation, the individual strings are then, if necessary, decoded, the objective function evaluated, a fitness value assigned to each individual and individuals selected for mating according to their fitness, and so the process continues through subsequent generations. In this way, the average performance of individuals in a population is expected to increase, as good individuals are preserved and bred with one another and the less fit individuals die out. The GA is terminated when some criteria are satisfied, e.g. a certain number of generations, a mean deviation in the population, or when a particular point in the search space is encountered.

4.3.1 Genetic Algorithms and the Traditional Methods

The working principle of GA clearly indicates that the GA significantly differs in some very fundamental way from other traditional search and optimization methods. The four major significant differences are:

- GAs search a population of solutions instead of a single point solution as in traditional search methods.
- GAs do not use derivative-based algorithm. It does not use any derivative information or other auxiliary knowledge; only the objective function and corresponding fitness levels influence the directions of search.
- GAs use probabilistic transition rules instead of deterministic principle.

It is important to note that the GA provides a number of potential solutions to a given problem and the choice of final solution is left to the user. In cases where a particular problem does not have one individual solution, for example a family of Pareto-optimal solutions, as is the case in multi-objective optimization and scheduling problems, then the GA is potentially useful for identifying these alternative solutions simultaneously.

4.3.2 Simple Genetic Algorithm

The schematic shown in Figure 4.12 illustrates the structure of a simple genetic algorithm (SGA) as described by Goldberg [8]. GA starts with initial random population consisted of potential solution points called individuals. The decision is made whether the individual is good or bad for the given problem based on the fitness obtained from the evaluation of objective function. Once the fitness value is evaluated and assigned to each individual, then initial population meets the first genetic operator, selection process. This operator provides more chances of survival for the strong individuals and to decay the weakest ones according to their fitness.

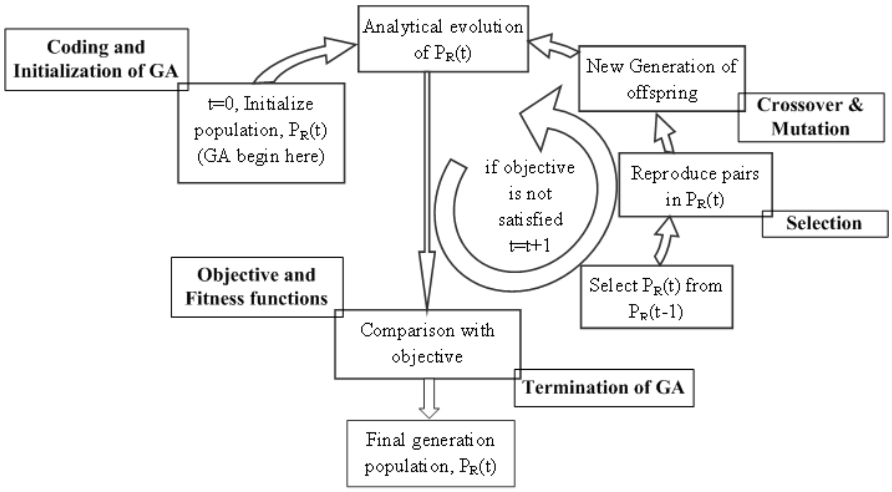

Fig. 4.12. A schematic representation of simple genetic algorithm outline population representation and initialization

Next, crossover operator is performed on selected individuals to build the new individuals by combining the existing ones. Crossover follows reproduction and allows two individuals to swap their structures depending on the probability factor. This result in the creation of a pair of offspring solution containing characteristics of their parents. Then the mutation operator is applied to supply diversity in the population. As the fitness of a population may remain static for a number of generations before a superior individual is found, the application of conventional termination criteria becomes problematic. A common practice is to terminate the GA after a pre-defined number of generations and then test the quality of the best members of the population against the problem definition. If no acceptable solutions are found, the GA may be restarted or a fresh search initiated with more number of generations.

4.3.2.1 Coding and Initialization of GA

4.3.2.1.1 Binary Coding

The most commonly used representation of chromosomes in the GA is that of the single-level binary string. Here, each decision variable (say, d and h) in the parameter set is encoded as a binary string (s_d and s_h, respectively) and these are concatenated to form a chromosome as shown in Figure 4.13. Binary-coded GAs are not restricted to use only integer and for the given lower bound (d_{min}) and upper bound (d_{max}) of a variable (say, d), the value of the variable (d) is calculated from the GA-string using the decoding scheme represented by Equation (4.12).

$$d = d^{min} + \frac{d^{max} - d^{min}}{2^{l_d} - 1} DV(s_d) \tag{4.12}$$

where l_d is the string length used to code the d variable and $DV(s_d)$ is the decoded value of the string s_d . This mapping function allows

$$\overset{S_h}{\underbrace{11001}_{h=25}}\overset{S_d}{\underbrace{01000}_{d=8}}$$

Fig. 4.13. A schematic representation of a chromosome with 5 bits for S_d and 5 bits for S_h

4.3.2.1.2 Real Coding

For a continuous search space, binary-coded GA faces many problems such as

- Hamming cliffs associated with certain strings (e.g., 01111 and 10000) from which a transition to a neighboring solution requires the alteration of many bits.
- Inability to achieve any arbitrary precision in the optimal solution. The more the required precision, the larger is the string length, results in more computational complexity.

In real coding, the variables are directly represented in real type as $\underbrace{25}_{h}\underbrace{8}_{d}$.

4.3.2.1.3 Initialization of GA

Initial population of a GA is normally determined at random. With a binary population of N_{ind} individuals whose chromosomes are L_{ind} bits long, $N_{ind} \times L_{ind}$ random numbers uniformly distributed from the set {0, 1} would be produced.

4.3.2.2 Objective Function and Fitness Function

The objective function is used to provide an analytical measure of how individuals have performed in the problem domain. The objective function, f is a function of decision variable(s). The fitness value of an individual/solution in the population is determined based on the fitness function which consists of the objective function value and the penalty value for constraint violation which is determined by a penalty function, $f_{costraint}$. Thus, the fitness value is calculated as follows:

Fitness value= f +$f_{costraint}$,

Since GAs mimic the survival-of-the-fittest of the nature to make a search process, GAs are suitable for maximization problem where objective function is directly used as fitness function F(x). In the case of a minimization problem, the fit individuals will have the lowest numerical value of the associated objective function. This situation is handled using a conversion of the objective function into an equivalent maximization problem and used as fitness function so that the optimum point remain unchanged.

4.3.2.3 Selection

Selection guides the tool to find the optimized solution by preferring individuals/members of the population with higher fitness over one with lower fitness. It is the operator which generates the mating pool. This operator determines that the number of times a particular individual will be used for reproduction and the number of offspring that an individual will produce. Some of the popularly used selection methods are as follows:

Roulette Wheel Selection Methods: Roulette wheel selection scheme chooses a certain individual with a probability proportional to its fitness.

$$p\left[I_{j,t}\right] = \frac{f\left(I_{j,t}\right)}{\sum_{k=1}^{n} f\left(I_{k,t}\right)} \tag{4.13}$$

where $p\left[I_{j,t}\right]$ is the probability of getting selected of any j^{th} individual at a generation t, $f\left(I_{j,t}\right)$ and $\sum_{k=1}^{n} f\left(I_{k,t}\right)$ are corresponding individual fitness and the sum of the fitness of the population with size n, respectively.

The property as represented by Equation (4.13) is satisfied by applying a random experiment that has some similarity with a generalized roulette game. In the roulette game, the slots are not equally wide that is, why different outcomes occur with different probabilities. Figure 4.14 gives a graphical representation of how this roulette wheel game works.

Linear rank selection: In this plan, a small group of individuals is taken from the population and the individual with best fitness is chosen for reproduction. The size of the group chosen is called the tournament size. A tournament size of two is called binary tournament.

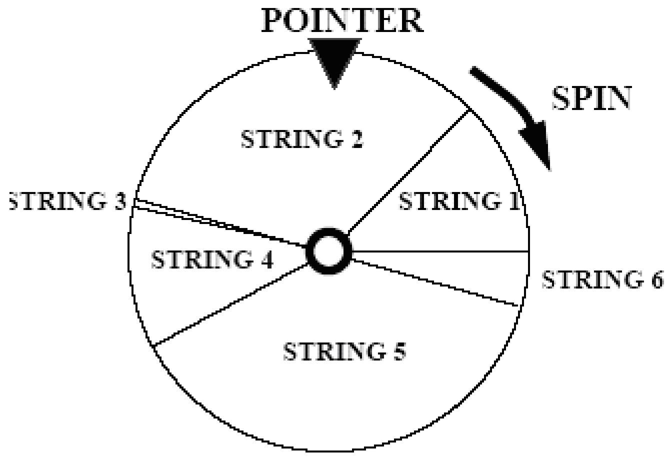

Fig. 4.14. Graphical representation of the Roulette wheel selection mechanism

In addition another scheme for selection is applied along with all three selection schemes discussed above which is called 'elitism'. The idea of elitism is to avoid the observed best-fitted individual dies out just by selecting it for the next generation without any random experiment. Elitism significantly influences the speed of the convergence of a GA. But it can lead to premature convergence also.

4.3.2.4 Crossover (Recombination)

The basic operator for producing new chromosomes in the GA is that of crossover. Like natural reproduction, crossover produces new individuals so that some genes of a new child come from one individual while others come from the other individual. In essence, crossover is the exchange of genes between the chromosomes of the two parents. The process may be described as cutting two strings at a randomly chosen position and swapping the two tails. It is known as the single-point crossover, and the mechanism is visualized in Figure 4.15. An integer position, i is selected at random with a uniform probability between one and the string length, l, minus one (i.e., $i \in [1, l-1]$). When the genetic information is exchanged among the parent individuals (represented by the strings, P_1 and P_2) about this point, two new offspring (represented by the strings, O_1 and O_2) are produced. The two offspring in Figure 4.15 are produced when the crossover point, i=4 is selected.

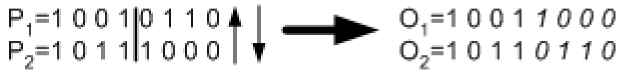

Fig. 4.15. A typical example of single point crossover

For multi-point crossover, multiple crossover positions (m) are chosen at random with no duplicates and sorted into ascending order. Then the bits between two successive crossover points are exchanged between the two parents to produce two new offspring. The process of multi-point crossover is illustrated in Figure 4.16 with shaded color.

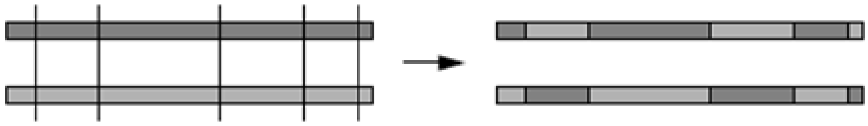

Fig. 4.16. Typical example of multi-point crossover (with m=5)

The idea behind multi-point crossover is that the parts of the chromosome representation that contributes to the most to the performance of a particular individual may not necessarily be contained in adjacent substrings. Further, multi-point crossover appears to encourage the exploration of the search space, thus making the search more robust.

4.3.2.5 Mutation

Mutation is nothing but deformation of the genetic information of an individual (solution) by means of some external influences. The bit-wise mutation operator changes a bit, 1 to 0, and vice versa, with a prescribed probability (called, mutation probability) as shown in Figure 4.17. In real reproduction, the probability that a certain gene is mutated is almost equal for all genes. So, it is near at hand to use the mutation technique for a given binary string, where there is a given probability that a single gene is modified. The probability should be rather low in order to avoid chaotic behavior of the GA.

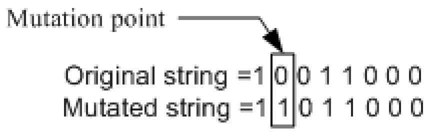

Fig. 4.17. Mutation effect on offspring's strings

4.3.2.6 Termination of the GA

Because the GA is a stochastic search method, it is difficult to formally specify convergence criteria. As the fitness of a population may remain static for a number of generations before a superior individual is found, the application of conventional termination criteria becomes problematic. A common practice is to terminate the GA after a pre-specified number of generations and then test the quality of the best members of the population against the problem definition. If no acceptable solutions are found, the GA may be restarted or a fresh search can be initiated. Attaining a pre-specified fitness function value or when best fitness of the population does not change appreciably over successive iterations may be also considered as a termination criteria.

4.3.3 Description of Working Principle of GA

The working principle of a (binary-coded) GA is described here with an engineering optimization problem, namely designing a can [9] as shown in Figure 4.18. The objective of the problem is to determine the optimum values of the diameter (d, in cm) and height (h, in cm) of the can in order to minimize its cost, f(d, h) subject to some constraints and the problem is defined as follows:

Minimize $$f(d,h)=c\left(\frac{\pi d^2}{2}+\pi dh\right),$$

Subject to

$$g(d,h)\equiv\frac{\pi dh^2}{4}\geq 300,$$
$$d_{min}\leq d\leq d_{max}$$
$$h_{min}\leq h\leq h_{max},$$

where c is the cost of the can material per square cm, which is taken as 0.005 and the minimum and maximum values of d and h are taken as, $d_{min}=h_{min}=0$ and $d_{max}=h_{max}=31$.

Therefore, in this problem the number of decision variables is two (d and h). The population size of GA is considered as six and it is kept constant throughout the GA-operation. GA operates in a number of iterations until a specified termination criterion is satisfied and in the Figure 4.18, the maximum number of iteration/generation (max_gen) is treated as the termination criteria.

GA iteration starts with the creation of six random solutions which are treated as the parent solutions. The chromosome structure of each solution is the same as presented in Figure 4.13. Then, the fitness values of all the parent solutions in the population are calculated using following fitness function (discussed in Section 4.3.2.2).

$$\text{Fitness value}=\begin{cases} f(d,h)+P_p\times[300-g(d,h)] & \text{if } g(d,h)\leq 300\\ f(d,h) & \text{if } g(d,h)>300\end{cases} \tag{4.14}$$

where P_p is the penalty parameter and the value of P_p is taken as 25. After that selection/reproduction (discussed in Section 4.3.2.3) operation is performed on the parent solutions based on the fitness values of solutions (as depicted in the binary tournament selection table in Figure 4.18) to form the mating pool. The number of solutions contained in the mating pool is to be equal to the number of parent

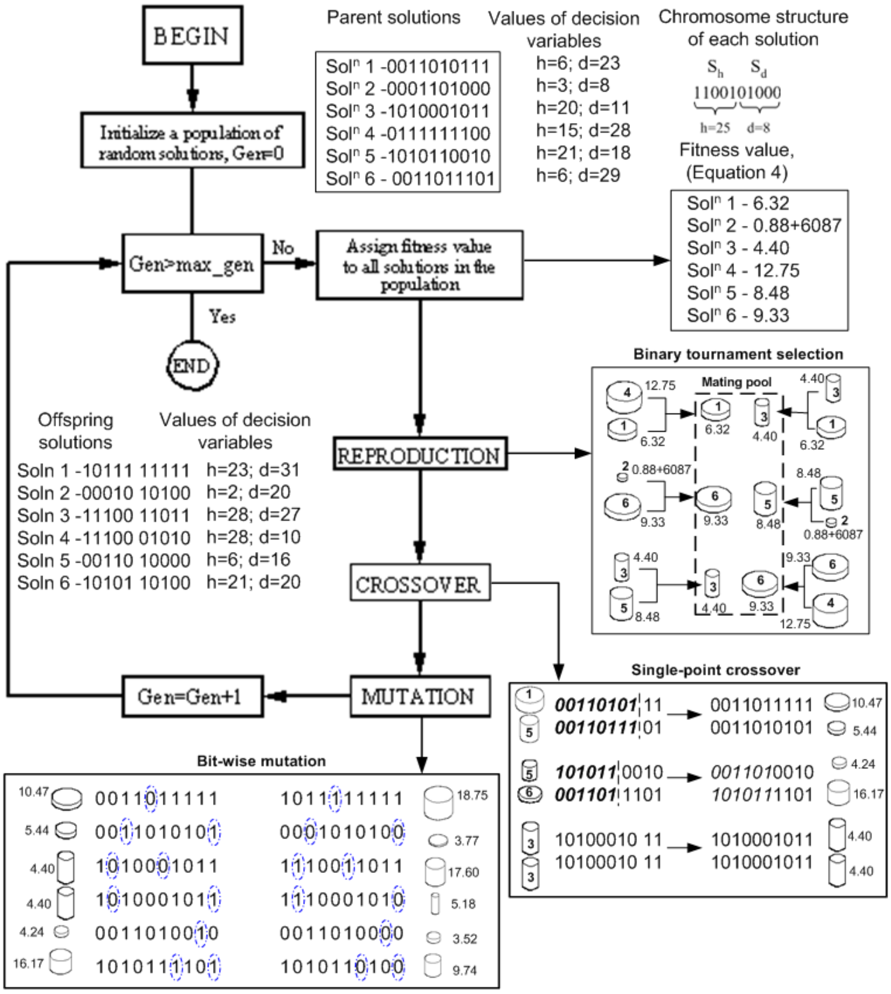

Fig. 4.18. Working principle of a genetic algorithm

solution in order to maintain a constant population size through out the GA-iteration. After that, crossover operator (discussed in Section 4.3.2.4) is applied among the two randomly chosen solutions from the mating pool based on a given probability (crossover-probability, say 0.9). Then, bit-wise mutation (discussed in Section 4.3.2.5) is carried out on each of the six solutions obtained after employing the crossover operator using a given mutation probability (say 0.02) and produces six new (offspring) solutions. The objective function values corresponding to each solution are depicted in the single-point crossover and bit-wise mutation tables in Figure 4.18. It completes one iteration/generation of GA. Now, the checking of GA-termination criterion is performed and if it satisfies the termination criteria, stop the GA-iteration process, otherwise start another generation/iteration with treating the six offspring solutions obtained in the previous iteration as the parent solutions and then assigning fitness value to all the solutions in the population and continue the same iteration procedure as described above.

4.4 Genetic Fuzzy Approaches

The performance of a FRBM depends on its knowledge base (KB) which consists of database (DB) (that is, information regarding membership functions) and rule base (RB). It is important to mention that determination of an appropriate knowledge base for a FRBM is not an easy task. The genetic algorithms (GAs) have been used by several investigators to design database and/or rule base of a FRBM. The fuzzy systems making use of a GA in their design process are called genetic-fuzzy systems (GFS). Figure 4.19 shows the schematic diagram of a genetic-fuzzy system, in which a GA-based learning/tuning based on example data (training data) is adopted, off-line to design the KB a FRBM. GA can also be used to tune the existing KB of a FRBM which may be designed based on the some common knowledge or expert information, to improve the performance of the existing FRBM. As the GA is found to be computationally expensive due to the nature of population-based optimization, the GA-based tuning is normally carried out off-line.

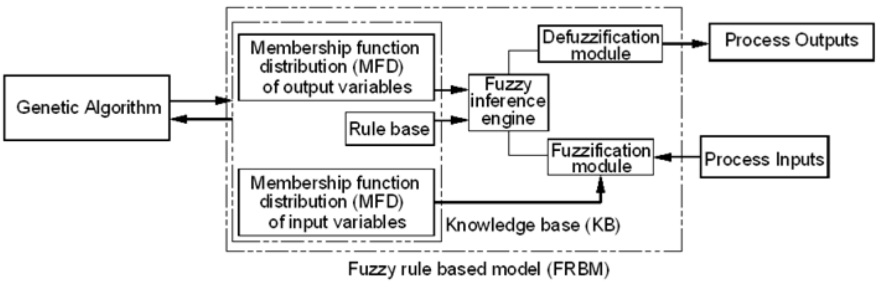

Fig. 4.19. Schematic layout of genetic-fuzzy system

A Mamdani-type fuzzy logic rule for a particular process having say, two input variables (x_1 and x_2) and one output variable (y) (each having triangular-type MFDs with 3 fuzzy subsets) may be expressed as

IF x_1 is A_1 AND x_2 is A_2 THEN y is B,

where A_1, A_2 and B are the fuzzy subsets (those can be expressed by suitable linguistic expression, such as LOW, MEDIUM, HIGH, etc.) of triangular-type membership function. In Section 4.2.2.2, it was discussed that in order to describe triangular-type MFDS with 3 fuzzy subsets one controlling parameter is required. A typical binary coded GA-string for optimizing the KB will look as shown in Figure 4.20.

$$\underbrace{011\ldots\ldots..1}_{\text{Rule base}}\underbrace{01..01}_{b_1}\underbrace{\ldots.11}_{b_2}\underbrace{01\ldots.1}_{b_3}$$

Fig. 4.20. A GA-string representing the rule base and the parameters related to membership functions of input-output variables a FRBM

where b_1, b_2 and b_3 are the (continuous) control (GA) variables related to MFDs corresponding to the two inputs and a single output variables x_1, x_2 and y, respectively. The number of bits used for optimizing the RB is equal to the number of maximum possible rules present in the RB. In this case, the number of rules will be $3 \times 3 = 9$, since each of the two input variables comprise of 3 fuzzy subsets. The information of b_1, b_2 and b_3 are coded by the next bits in GA-string.

There are, in fact, three different approaches of designing genetic-fuzzy system (GFS), according to the KB components including in the learning process. These are as follows

Genetic Learning /Tuning of the Fuzzy Logic Controller Data Base

Here, the GA is used to optimize the appropriate value(s) of the controlling parameter(s) that define the typical type of MFDs. In other words, for examples in case of triangular type MFDs, it is used to move and to expand or shrink the base width(s) ($b_1/b_2/b_3$) of each interior isosceles triangle. The extreme triangles will be right triangles and the GA will make it either bigger or smaller. During GA-based optimization the parameters (b_1, b_2 and b_3) are allowed to vary in a range specified by the designer. Thus, in this approach through evolution, the GA will find a good database for the FRBM but the RB will be kept as same what was initially considered by the designer. In this case, the GA-string shown in Figure 4.21 will look as follows:

$$\underbrace{01..01}_{b_1}\underbrace{\ldots.11}_{b_2}\underbrace{01\ldots.1}_{b_3}$$

Fig. 4.21. A GA-string representing the parameters related to membership functions of input-output variables of a FRBM

Genetic Learning/Tuning of the Fuzzy Rule Base

All methods belonging to this family assume the existence of a pre-defined DB for the FLC. In one method, an initial user-defined rule base (assignment of fuzzy subset to the output variable of each rule) constructed based on designer's experience of the process to be controlled is tuned using GA. In GA-sting (as shown in Figure 4.22), each rule is represented by a single bit (1 or 0), where 1 and 0 respectively indicate the presence or absence of the rule in the optimum rule base.

$$\underbrace{011\ldots\ldots01}_{\text{Rule base}}$$

Fig. 4.22. A GA-string representing the rule base of a FRBM

In the method where the RB is designed using GA automatically, additional bits will be included in the GA-string demonstrated in Figure 4.22 (as shown in Figure 4.23) [10]. For the case where output variable is considered to have 2 fuzzy subsets, one bit will be required in order to determine the fuzzy subset to the output variable for each rule (e.g., 0 for LOW and 1 for MEDIUM). Two bits required for the case where output variable is considered to have 3/4 fuzzy subsets (e.g., 00 for LOW, 01 for MEDIUM, 10 for HIGH and 11 for VERY HIGH), and so on.

$$\underbrace{011\ldots\ldots1}_{\text{Rule base}}\underbrace{01110\ldots\ldots\ldots\ldots01}_{\text{Automated rule development}}$$

Fig. 4.23. A GA-string representing the rule base and automated rule development of a FRBM

Genetic Learning/Tuning of the Fuzzy Knowledge Base
In this approach both the RB and DB are designed/optimized using GA as stated above simultaneously and the corresponding GA-string will look as shown in Figure 4.24.

$$\underbrace{011\ldots\ldots1}_{\text{Rule base}}\underbrace{01..0}_{b_1}\underbrace{1....11}_{b_2}\underbrace{01....1}_{b_3}\underbrace{01001101\ldots\ldots11}_{\text{Automated rule development}}$$

Fig. 4.24. A GA-string representing the rule base with automated rule development and the parameters related to membership functions of input-output variables of a FRBM

Besides the way how to construct/design the KB of FLC, selection of the appropriate shape of fuzzy subsets/membership function distributions (MFDs) for both the input and output variables in case of Mamdani-type FLC and selection of shapes of fuzzy subsets of input variables as well as the appropriate structure(s) of rule output/consequent function(s) and determination of optimal values of coefficients and power terms of rule consequent functions are important issues. In order to overcome these problems, a rigorous study with different choices is required in order to obtain a good model for a manufacturing process.

GA is also used in the genetic-fuzzy system where the TSK-type fuzzy logic rules (as defined in Section 4.2.11.2.1) are employed. Genetic Linear Regression (GLR) approach [11] is one of the most popular approaches in designing the KB of TSK-type FRBM. In genetic linear regression (GLR) approach, the GA is introduced partly in the multiple regression method. The GLR method will take the advantages of both the regression technique and GA, and have a capability to find global optimum and good convergence properties. In this approach, the KB of the FRBM is optimized using the combined method of linear regression (LR) approach and genetic algorithm. In this method the coefficients of output function of each rule are determined using a linear regression approach whereas the input variables' exponential parameters are simultaneously optimized using a GA. In addition to that, the MFDs also tune using a GA. In order to accomplish this, besides the exponential parameters of input variables, the controlling parameter(s) describing the MFDs (as discussed in Section 4.2.2.2) are considered as GA-variables. The working principle of GLR approach (as illustrated in Figure 4.25) consists of following five major steps:

- Step-I: Set an initial set (population) of values of power terms of a given regression function at random
- Step-II: Evaluate the function coefficients based on least square method
- Step-III: Checking of fitness value (if satisfied terminate the iteration procedure)
- Step-IV: Update the values of power terms of regression function using GA-operators
- Step-V: Repeat Step-II

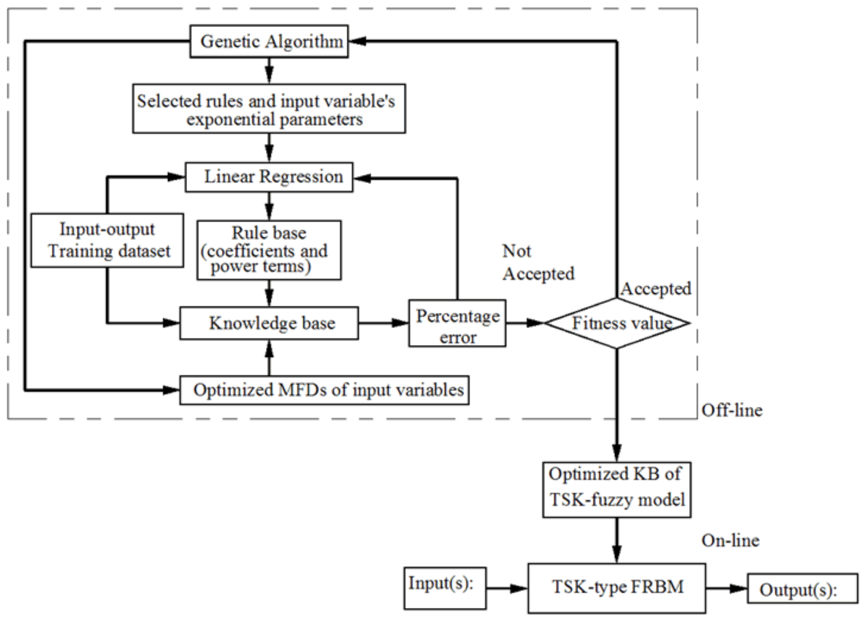

Fig. 4.25. Flow chart of genetic linear regression approach to construct KB of a FRBM

The GA-string of GLR approach will look as presented in Figure 4.26.

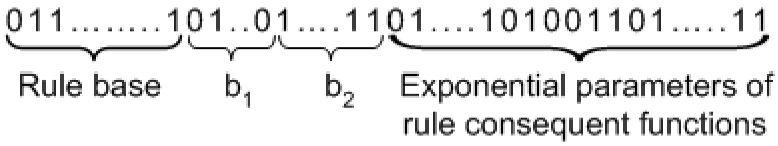

Fig. 4.26. A GA-string representing the rule base, parameters related to membership functions of input variables and exponential parameters of rule consequent functions of a TSK-type FRBM

The proposed GLR has an added facility to carryout the task of tuning MFDs of input variables simultaneously in the same framework of GA.

In order to determine the coefficients of the output functions of TSK-type fuzzy rules, a general expression of multiple linear regression system with TSK-fuzzy model is derived as follows. The Equation (4.11) may be rewritten by denoting $\prod_{v=1}^{n} \mu_V(x_{1,\ldots,n}) = \eta_r$ for simplicity, in the following form:

$$Y = F(x_1, .. x_n)$$

$$= \frac{\sum_{r=1}^{R_f^l} \eta_r\left(a_1^r f_1^r(x_{1,\ldots,n}) + a_2^r f_2^r(x_{1,\ldots,n}) + \ldots\ldots\ldots.. + a_k^r f_k^r(x_{1,\ldots,n})\right)}{\sum_{r=1}^{R_f^l} \eta_r}$$

$$= \frac{\begin{array}{l}\eta_1\left(a_1^1 f_1^1(x_{1,..,n}) + .. + a_k^1 f_k^1(x_{1,..,n})\right) + .. \\ + \eta_{l_{tr}}\left(a_1^{tr} f_1^{tr}(x_{1,\ldots,n}) + .. + a_{tj}^{tr} f_{tj}^{tr}(x_{1,\ldots,n}) + .. + a_k^2 f_k^2(x_{1,..,n})\right) + . \\ . + \eta_{R^l}\left(a_1^{R_f^l} f_1^{R_f^l}(x_{1,..,n}) + .. + a_k^{R_f^l} f_k^{R_f^l}(x_{1,..,n})\right)\end{array}}{\eta_1 + \eta_2 + \ldots\ldots.. + \eta_{R_f^l}} \qquad (4.15)$$

Let us assume we have a set of input-output tuple (D) of S number of sample data where the output $y^{(i)}$ is assigned to the input $(x_1^{(i)}, x_2^{(i)}, \ldots\ldots\ldots, x_n^{(s)})$

$$D = \left\{\left(x_1^{(1)}, \ldots\ldots, x_n^{(1)}, y^{(1)}\right), \left(x_1^{(2)}, \ldots\ldots, x_n^{(2)}, y^{(2)}\right) \ldots\ldots\ldots.\left(x_1^{(s)}, \ldots\ldots, x_n^{(s)}, y^{(s)}\right)\right\}$$

Now, the total quadratic error that is caused by the TSK-type FLC with respect to the given data set:

$$E = \sum_{l=1}^{s}\left(f\left(x_1^{(l)}, x_2^{(l)}, \ldots. x_n^{(l)}\right) - y^{(l)}\right)^2 \qquad (4.16)$$

In order to minimize E, we have to choose the parameters, $\left\{\left(a_1^1, \ldots., a_k^1\right), \left(a_1^2, \ldots.. a_k^2\right), \ldots\ldots, \left(a_1^{R_f}, \ldots., a_k^{R_f}\right)\right\}$ appropriately.

where the parameter a_j^r indicates the j_{th} coefficient of the output function of r_{th} rule.

To determine the above parameters, we take the partial derivatives of E with respect to each parameter and require them to be zero, i.e., $\frac{\partial E}{\partial a_j^r} = 0$, where $j = \{1,2,....,k\}$ and $r = \{1,2,.....,R_f\}$.

Now, we obtain the partial derivation of E with respect to the parameter $a_{t_j}^{t_r}$,

$$\frac{\partial E}{\partial a_{t_j}^{t_r}} = \sum_{l=1}^{S} 2 \cdot \left(f\left(x_1^{(l)},........,x_n^{(l)}\right) - y^{(l)}\right) \cdot \frac{\partial f\left(x_1^{(l)},........,x_n^{(l)}\right)}{\partial a_{t_j}^{t_r}}$$

$$= 2 \cdot \sum_{l=1}^{S} \left(\frac{\sum_{r=1}^{R_f^l} \eta_r \left(a_1^r f_1^r\left(x_{1,..,n}^l\right) + ... + a_k^r f_k^r\left(x_{1,..,n}^l\right)\right)}{\sum_{r=1}^{R_f^l} \eta_r} - y^l \right) \cdot \frac{\eta_{t_r} \cdot f_{t_j}^{t_r}\left(x_{1,..,n}^l\right)}{\sum_{r=1}^{R_f^l} \eta_r}$$

$$= 2\left(\left(\frac{\sum_{l=1}^{S}\sum_{r=1}^{R_f^l} \eta_r \left(a_1^r\right) f_1^r\left(x_{1,..,n}^l\right) \eta_{t_r} f_{t_j}^{t_r}\left(x_{1,..,n}^l\right)}{\left(\sum_{r=1}^{R_f^l} \eta_r\right)^2}\right) + .. + \left(\frac{\sum_{l=1}^{S}\sum_{r=1}^{R_f^l} \eta_r \left(a_k^r\right) f_k^r\left(x_{1,..,n}^l\right) \eta_{t_r} f_{t_j}^{t_r}\left(x_{1,..,n}^l\right)}{\left(\sum_{r=1}^{R_f^l} \eta_r\right)^2}\right)\right)$$

$$- 2\left(\frac{\sum_{l=1}^{S} y^l \cdot \eta_{t_r} f_{t_j}^{t_r}\left(x_{1,..,n}^l\right)}{\sum_{r=1}^{R_f^l} \eta_r}\right) = 0, \tag{4.17}$$

Thus, the Equation (4.16) provides the following system of linear equations from which we can compute the coefficients $\left\{\left(a_1^1,...., a_k^1\right), \left(a_1^2,..... a_k^2\right),......, \left(a_1^{R_f},...., a_k^{R_f}\right)\right\}$:

$$\sum_{r=1}^{R_f} \sum_{j=1}^{K} a_j^r \sum_{l=1}^{S} \frac{\prod_{v=1}^{n} \mu_v^r\left(x_{1,..,n}^l\right)}{\left(\sum_{r=1}^{R_f^l} \prod_{v=1}^{n} \mu_v^r\left(x_{1,..,n}^l\right)\right)^2} f_j^r\left(x_{1,..,n}^l\right) \cdot f_{t_j}^{t_r}\left(x_{1,..,n}^l\right) \cdot \prod_{v=1}^{n} \mu_v^{t_r}\left(x_{1,..,n}^l\right)$$

$$= \sum_{l=1}^{S} \frac{y^l \prod_{v=1}^{n} \mu_v^{t_r}\left(x_{1,..,n}^l\right)}{\sum_{r=1}^{R_f^l} \prod_{v=1}^{n} \mu_v^r\left(x_{1,..,n}^l\right)} f_{t_j}^{t_r}\left(x_{1,..,n}^l\right) \tag{4.18}$$

In matrix form, Equation (4.18) will be written as:

$$\begin{bmatrix} \alpha^r_{11} & \alpha^r_{12} & \cdot & \alpha^r_{1K} \\ \alpha^r_{21} & \alpha^r_{22} & \cdot & \alpha^r_{2K} \\ \cdot & \cdot & \cdot & \cdot \\ \alpha^r_{K1} & \alpha^r_{K2} & \cdot & \alpha^r_{KK} \end{bmatrix} \begin{bmatrix} a^r_1 \\ a^r_{21} \\ \cdot \\ a^r_K \end{bmatrix} = \begin{bmatrix} \beta^r_1 \\ \beta^r_2 \\ \cdot \\ \beta^r_K \end{bmatrix} \tag{4.19}$$

where $\alpha^r_{tj} = \sum_{l=1}^{S} f^r_j(x^l_1, x^l_2 .., x^l_n) f^r_t(x^l_1, x^l_2 .., x^l_n)$; $\beta^r_t = \sum_{l=1}^{S} y^l f^r_t$,

where s is the number of training (input-output) sample data. Thus, Equation (4.18) provides a solution of the function coefficients (a^r_j) of the TSK-type fuzzy rule consequents for a given value of the input variable's exponential terms. To solve the Equation (4.18) in order to find the values of function coefficients, the Gauss Algorithm with Column Pivot Search Method is used here. More generally, any conventional numerical method, which provides a representative solution of Equation (4.18), may be adopted.

4.5 Application to Modeling of Machining Process

Numerous works on modeling of machining processes using soft computing tools including different GA-fuzzy approaches can be found in the review paper [12]. In this chapter, some of the previous research works of the author on modeling of manufactuing processes using GA-fuzzy approaches are presented.

4.5.1 Modeling Power Requirement and Surface Roughness in Plunge Grinding Process

In [13], a study was carried out to model power requirement and surface roughness in plunge grinding process using a GA-fuzzy approach. The model considers the input variables such as wheel speed, work speed and feed rate those mainly influence the power requirement and surface roughness obtained on the grind surface. In this study the main objective was to find the effect of various types of MFs considered for input-output variables on the performances of Mamdani-type FRBMs. The approach of tuned rule base and MFs simultaneously using GA was adopted. The performance of the models was tested (as shown in Figure 4.27) with experimental results considering 52100 steel as work material and D126K5V as grinding wheel specification. A digital clamp power meter is used to take the measurements of power requirement in grinding and the surface roughness is measured with the help of a perthometer (S6R). In Figure 4.27, Error 1, Error 2 and Error 3 are the percentage deviations of results predicted by FRBMs with triangular, 2nd order polynomial and 3rd order polynomial-type MFDs, respectively from that obtained in experimentation. It is observed that higher order polynomial type MFs showed better results. It may happen because the input-output relationship in grinding is highly nonlinear and the linear MFDs (triangular type) may not be sufficient.

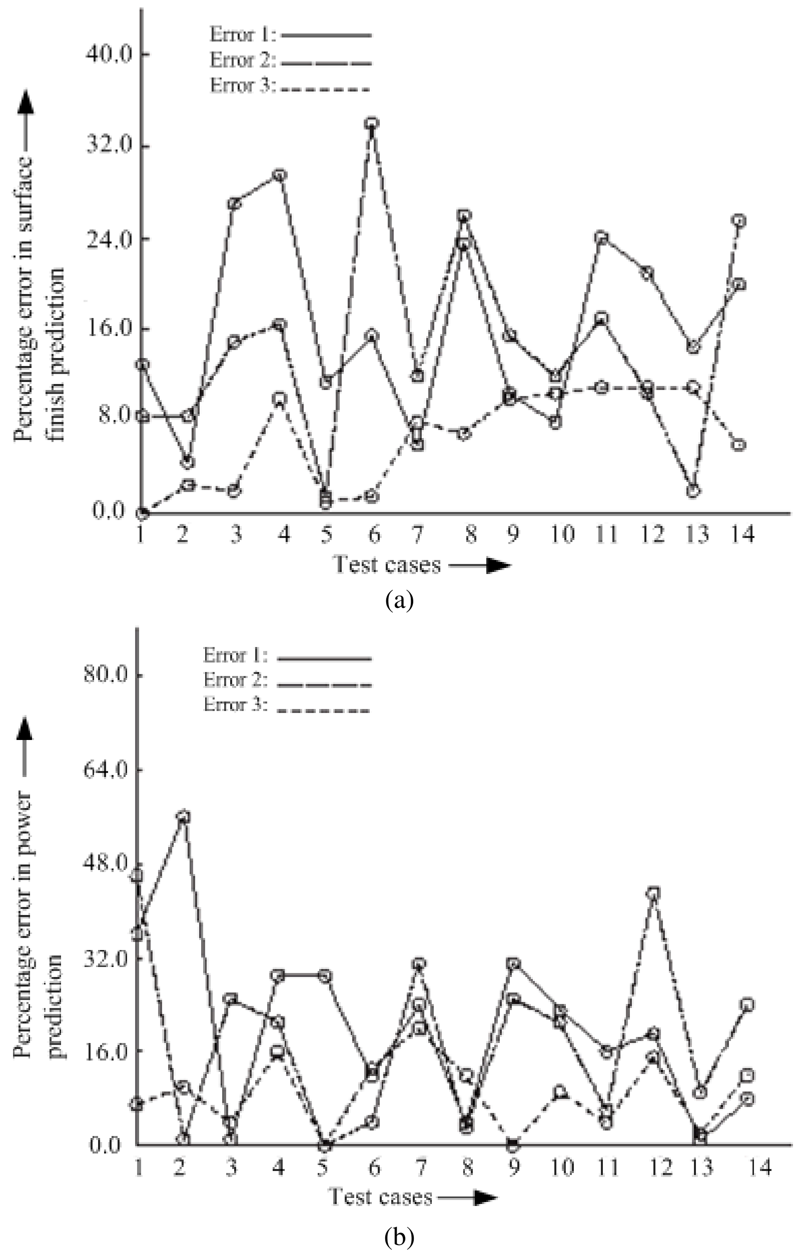

Fig. 4.27. Comparison of performances of FRBMs (with different types MFs) with those of experimental results (a) surface roughness (b) power requirement

In the above study, the GA was used to optimize the manually-defined KB of the FRBM. The manually-defined KB is designed based on the expert's knowledge of the process that may not be complete. Sometimes, it becomes difficult to gather knowledge of the process beforehand. To overcome this difficulty, the method for automatic design of fuzzy KB is adopted to model power requirement and surface roughness in plunge grinding process [10]. Table 4.1 describes the comparative results of root mean square (RMS) percentage deviations exhibited by fuzzy rule-based models (FRBMs) from those of real experimental values. It is found that the approach of automatic design of RB and tuning of MFs simultaneously using GA provides better result over the approach of tuned manually constructed RB and MFs simultaneously using GA. It happens because all the manually-designed fuzzy rules may not be good, whereas the GA has the capability of finding the good fuzzy rules through extensive search. Moreover, the main disadvantage of using later approach (the approach of tuned manually constructed RB and MFs) lies in the fact that the designer is required to have a thorough knowledge of the process to be controlled. Thus, a considerable amount of time is spent on manual construction of fuzzy RB. In the approach of automatic design of RB and tuning of MFs simultaneously using GA, no effort is made for designing the fuzzy rule-base manually and a good KB of the FRBM is designed automatically using a GA from a set of example (training) data.

Table 4.1. Comparison of RMS percentage deviations exhibited by fuzzy rule-based models from those of real experimental values

	Power requirement		
	Mathematical model	FRBM based on the approach of tuned rule base and MFs simultaneously using GA (approach 2)	FRBM based on the approach of automatic design of rule base and tuning of MFs simultaneously using GA (approach 1)
RMS percentage error	31.51	8.13	5.34
	Surface roughness		
RMS percentage error	16.44	10.22	6.32

4.5.2 Study of Drilling Performances with Minimum Quantity of Lubricant [14]

The main objective of this study is to investigate the performances of FRBMs based on Mamdani-type and TSK-type of fuzzy logic rules, and different shapes (namely, 2nd order polynomial and trapezoidal types) of MFs for prediction and performance analysis of machining with minimum quantity of lubricant (MQL) in drilling of Aluminium (AA1050). In this study, predictions of surface roughness obtained in drilling and the corresponding cutting power and specific cutting force requirements for different amounts of lubricant rate will be carried out through a comparative analysis of the results of models with experimental results as well as those published in the literature. The approach of tuned RB and MFs simultaneously using GA and genetic linear regression method was adopted to construct the KB of Mamdani-type and TSK-type FRBM, respectively. The structure of rule-consequent function for TSK-type fuzzy rules is used as follows

$$y = c_1 V_c^{p_1} + c_2 F_r^{p_2} + c_3 L_r^{p_3} \tag{4.20}$$

where c_1, c_2 and c_3 are the function coefficients and p_1, p_2 and p_3 are the exponential parameters of rule consequent function. V_c, F_r and L_r are the input variables, cutting speed, feed rate and rate of lubricant, respectively.

A helical K10 drill (R415.5-0500-30) was manufactured according to DIN6537 by Sandvik(R). The drill has a point angle of 140º, 28 mm of flute length and is of 10% cobalt grade. The drills possess a diameter of 5 mm and are coated with TiAlN. A Kistler® piezoelectric dynamometer 9272 with a load amplifier was used to acquire the torque and the feed force. Data acquisitions were made through piezoelectric dynamometer by interfacing RS-232 to load amplifier and PC using the appropriate software, Dynoware Kistler(R). The surface roughness was evaluated (R_a according to ISO 4287/1) with a Hommeltester T1000 profilometer.

Here, 4 different models related to surface roughness and, other 4 different models for cutting power/specific cutting force requirement) are developed. The four different FRBMs are constructed using two different types of fuzzy logic rules (Mamdani-type and TSK-type) and two different shapes of MFs.

The comparative results of surface roughness, cutting power and specific cutting force with different lubricant flow rates for different cutting speed and feed rate are described in Figure 4.28, Figure 4.29 and Figure 4.30, respectively. In this study, nine different cases (of cutting speed and feed rate) are considered based on which the effects of lubricant flow rate on machining performances are analyzed. In these Figures, Model I indicates FRBM with Mamdani-type FLR and 2nd order polynomial MFs, Model II represents FRBM with Mamdani-type FLR and trapezoidal MFs, Model III shows FRBM with TSK-type FLR and 2nd order polynomial MFs and Model IV indicates FRBM with TSK-type FLR and trapezoidal MFs. In the following subsections the evolutions of the prediction performances of these models toward the effects of surface roughness, cutting power and specific cutting force with lubrication rate are discussed.

Surface roughness

In Figure 4.28, the predicted values of surface roughness by FRBMs are compared with the experimental values for 9 cases (Figure 4.28(i) to Figure 4.28(ix)). It has been observed that the performance of Model II is better than Model I for first 5 cases and case number 6. For other cases Model I outperforms over model II. But, the consistencies of deviations of results from that of the experimental values are not good for both the Model I and Model II in all the cases. In contrast, the results of Model III shows better than Model I and Model II for some cases (Figure 4.28 (iv), (v) and (vii)), but for other cases that are deteriorated compared to Model I and Model II. On the other hand it is noticed that the results of Model IV (FRBM with TSK-type FLR and trapezoidal MFs) yield less error (deviation from the

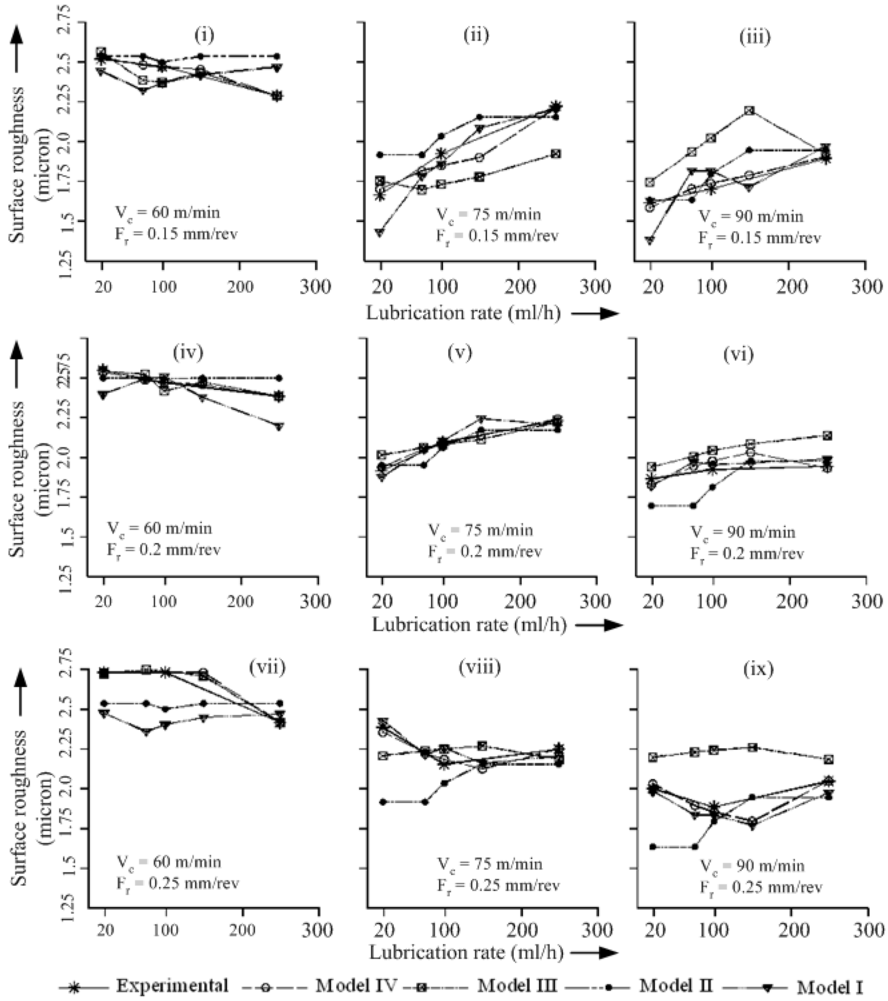

Fig. 4.28. Comparative results of surface roughness with different lubricant flow rates for different cutting speed and feed rate

experimental values) in majority than that of the other models for all (9) cases. Furthermore, it is noticed that the results of Model IV are also consistent for different values of lubrication rate. The maximum value of percentage error exhibited by Model IV is 2.2428, which is well accepted in industrial practice.

By analyzing the experimental values as well as results obtained by Model IV, it has been observed that the surface roughness is improved by increasing the flow rate for lower values of cutting speed (60 m/min) and constant feed rate. But, for higher values of cutting speed, the surface roughness deteriorated with increasing flow rate (75 and 90 m/min). In contrast, for a constant cutting speed, the rate of change of surface quality with flow rate is minimized as feed rate increase.

Cutting power

By analyzing the results of various models and experimental values as depicted in Figure 4.29, it has been observed that Model I as well as Model II provides poor results than other two models (Model III and Model IV). In contrast, it is found that both the models, Model III and Model IV obtain the best performance for predicting cutting power with the quantity of lubricant for a given cutting speed and feed rate. However, in cases V_c=90; f=0.15 and V_c=90; f=0.25, the Model IV shows better results than Model III.

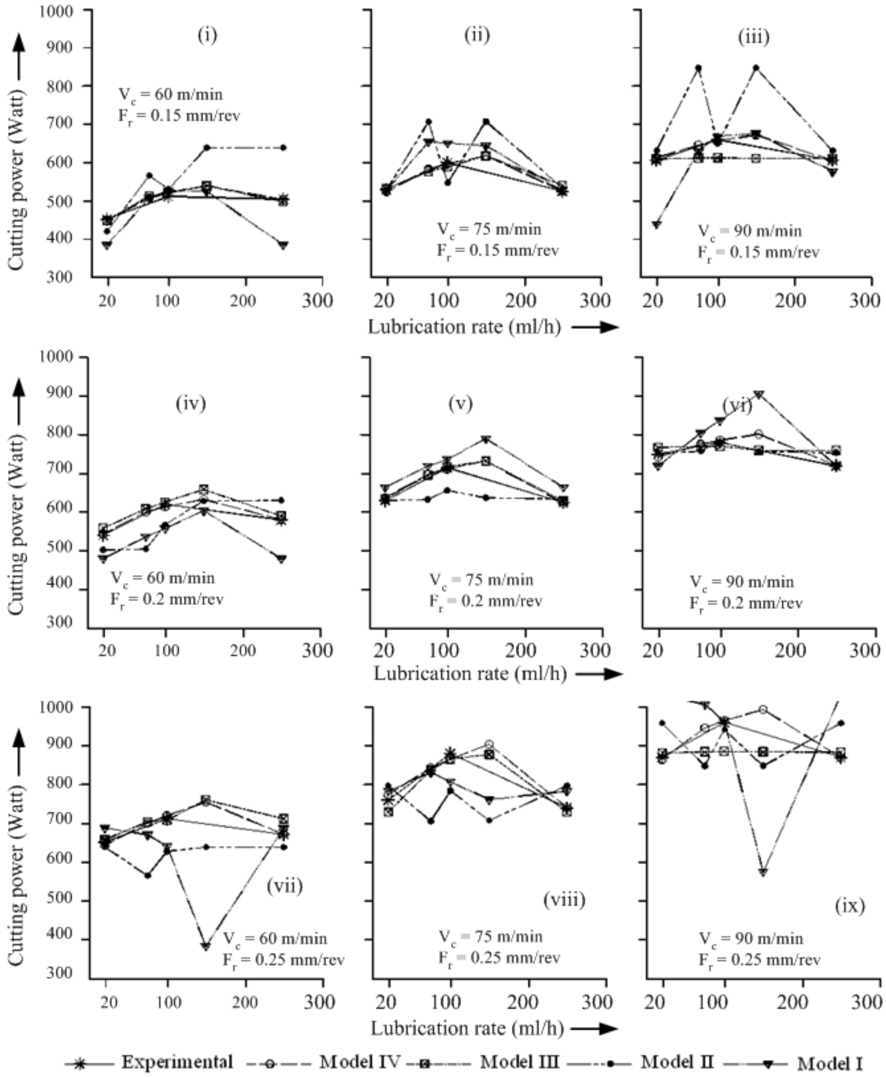

Fig. 4.29. Comparative results of cutting power with different lubricant flow rates for different cutting speed and feed rate

By analyzing the experimental values as well as results obtained by Model III and Model IV, it has been revealed that for a fixed value of cutting speed and feed rate, the cutting power increase to a certain value of lubrication flow rate. After that the value of cutting power decreases with increasing flow rate. From Figure 4.29, it is found that for constant cutting speed, the cutting power requirement increase with feed rate. It is also observed that the value of cutting power increases with cutting speed when feed rate is kept as a constant value.

Specific cutting force

As like cutting power, here also Model III and Model IV show the best result in predicting specific cutting force with the quantity of lubricant for a given cutting speed and feed rate (Figure 4.30). This is because; both the cutting power and specific force are depended on the same parameter, torque and a linear relationship is maintained among them. The variation of specific cutting force requirement with lubricant flow rate and other input parameters, cutting power and feed rate exhibit the same phenomena as found in cutting power.

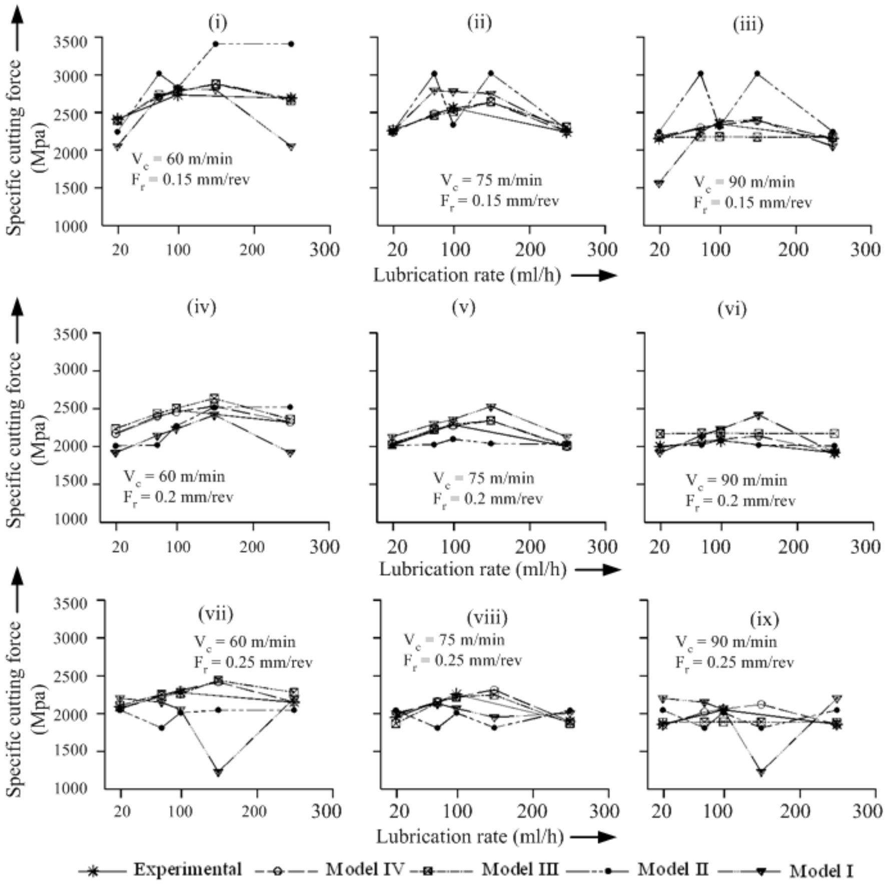

Fig. 4.30. Comparative results of specific cutting force with different lubricant flow rates for different cutting speed and feed rate

From above discussions, it may be pointed out that FRBMs with TSK type fuzzy logic rules provide best result in predicting surface roughness, cutting power and specific cutting force. Specifically for surface roughness, trapezoidal MFs is well suited, while trapezoidal as well as second order polynomial MFs give almost similar performances in predicting cutting power/specific cutting force requirements in drilling of Aluminium AA1050 with emulsion with oil Microtrend 231L lubricant. The above techniques may be adopted for developing FRBMs for other machining (drilling) performance parameters. Once the model is developed, it may be used on-line in drilling machine to control the MQL as per desired outputs.

References

[1] Groover, M.: Automation, Production System, and Computer Integrated Manufacturing. Prentice-Hall Int'l, Upper Saddle River (2001)

[2] Kosko, B.: Neural Network and Fuzzy Systems. Prentice-Hall, New Delhi (1994)

[3] Zadeh, L.A.: Fuzzy sets. Information and Control 8(3), 338–353 (1965)

[4] Mamdani, E.H., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machine Studies 7(1), 1–13 (1975)

[5] Sugeno, M., Kang, G.T.: Structure identification of fuzzy model. Fuzzy Sets and Systems 28(1), 15–33 (1988)

[6] Tsukamoto, Y.: Fuzzy information theory. Daigaku Kyoiku Pub. (2004)

[7] Takagi, T., Sugeno, M.: Fuzzy identification of systems and its application to modeling and control. IEEE Transactions on Systems, Man, and Cybernetics 15(1), 116–132 (1985)

[8] Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading (1989)

[9] Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. John Wiley & Sons Ltd, England (2001)

[10] Nandi, A.K., Pratihar, D.K.: Automatic Design of Fuzzy Logic Controller Using a Genetic Algorithm – to Predict Power Requirement and Surface finish in Grinding. Journal of Material Processing Technology 148(3), 288–300 (2004)

[11] Nandi, A.K.: TSK-Type FLC using a combined LR and GA: surface roughness prediction in ultraprecision turning. Journal of Material Processing Technology 178(1-3), 200–210 (2006)

[12] Chandrasekaran, M., Muralidhar, M., Krishna, C.M., Dixit, U.S.: Application of soft computing techniques in machining performance prediction and optimization: a literature review. Int. J. Advance Manufacturing Technology 46, 445–464 (2010)

[13] Nandi, A.K., Pratihar, D.K.: Design of a Genetic-Fuzzy System to Predict Surface finish and Power Requirement in Grinding. Fuzzy Sets and Systems 148(3), 87–504 (2004)

[14] Nandi, A.K., Davim, J.P.: A Study of drilling performances with Minimum Quantity of Lubricant using Fuzzy Logic Rules. Mechatronics 19(2), 218–232 (2009)